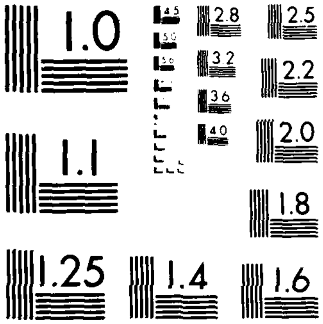
AD-A200 436 AERODYNAMIC PERFORMANCE OF WINGS OF ARBITRARY PLANFORM 1/3
IN INVISCID INCOMPRESSIBLE IRROTATIONAL FLOW(U) NAVAL
POSTGRADUATE SCHOOL MONTEREY CA C L HOLM SEP 88

UNCLASSIFIED F/G 1/1 NL

UNCLA



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

AERODYNAMIC PERFORMANCE OF WINGS
OF ARBITRARY PLANFORM IN
INVISCID, INCOMPRESSIBLE, IRROTATIONAL
FLOW

by

Chris L. Holm

September 1988

Thesis Advisor Theodore H. Gawain

Approved for public release; distribution is unlimited.

Unclassified

security classification of this page

REPORT DOCUMENTATION PAGE

1a Report Security Classification Unclassified		1b Restrictive Markings
2a Security Classification Authority		3 Distribution Availability of Report Approved for public release: distribution is unlimited.
2b Declassification Downgrading Schedule		
4 Performing Organization Report Number(s)		5 Monitoring Organization Report Number(s)
6a Name of Performing Organization Naval Postgraduate School	6b Office Symbol *(if applicable)* 31	7a Name of Monitoring Organization Naval Postgraduate School
6c Address *(city, state, and ZIP code)* Monterey, CA 93943-5000		7b Address *(city, state, and ZIP code)* Monterey, CA 93943-5000
8a Name of Funding Sponsoring Organization	8b Office Symbol *(if applicable)*	9 Procurement Instrument Identification Number
8c Address *(city, state, and ZIP code)*		10 Source of Funding Numbers Program Element No | Project No | Task No | Work Unit Accession No

11 Title *(include security classification)* AERODYNAMIC PERFORMANCE OF WINGS OF ARBITRARY PLANFORM IN INVISCID, INCOMPRESSIBLE, IRROTATIONAL FLOW

12 Personal Author(s) Chris L. Holm

13a Type of Report Master's Thesis	13b Time Covered From To	14 Date of Report *(year, month, day)* September 1988	15 Page Count 201

16 Supplementary Notation The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

17 Cosati Codes			18 Subject Terms *(continue on reverse if necessary and identify by block number)*
Field	Group	Subgroup	computational fluid dynamics,horse shoe vortex, circulation

19 Abstract *(continue on reverse if necessary and identify by block number)*

This thesis contains discussion, theory and program code for a computational fluid dynamics (CFD) model of a wing of arbitrary planform. The model assumes incompressible, inviscid, irrotational flow. The program computes forces acting on the wing by modeling the flow with a set of horse shoe vortex elements. It models the flow over an arbitrary wing using two solutions. One solution is the ideal lift, associated with a cambered and twisted wing. The other solution is the additional lift associated with a flat wing. The program computes wing camber and twist using an elliptic loading distribution. The thesis includes the FORTRAN source code, a separable User's Manual for the VORTEX program, discussion of the theory applied in the model, and instructions for operating the program. It shows a sample wing planform with tabular and graphic results.

The thesis also discusses two other CFD models based on circulation (Γ) and pressure difference (ΔC_p). It presents some of the problems and solutions in grid generation.

20 Distribution Availability of Abstract ☒ unclassified unlimited □ same as report □ DTIC users		21 Abstract Security Classification Unclassified
22a Name of Responsible Individual Theodore H. Gawain	22b Telephone *(include Area code)* (408) 646-2808	22c Office Symbol 67Gn

DD FORM 1473,84 MAR

83 APR edition may be used until exhausted
All other editions are obsolete

security classification of this page

Unclassified

Aerodynamic Performance of Wings
# of Arbitrary Planform in
Inviscid, Incompressible, Irrotational Flow

by

Chris L. Holm
Lieutenant Commander, United States Navy
B.A., University of New Mexico, 1975

Submitted in partial fulfillment of the
requirements for the degrees of

MASTER OF SCIENCE IN AERONAUTICAL ENGINEERING
and
AERONAUTICAL ENGINEER

from the

NAVAL POSTGRADUATE SCHOOL
September 1988

Author: ______________________________
Chris L. Holm

Approved by: Theodore H Gawain
Theodore H. Gawain, Thesis Advisor

Richard M. Howard
Richard Howard, Second Reader

Max F Platzer
Max F. Platzer, Chairman,
Department of Aeronautics and Astronautics

G E Schacher
Gordon E. Schacher,
Dean of Science and Engineering

ABSTRACT

This thesis contains discussion, theory and program code for a computational fluid dynamics (CFD) model of a wing of arbitrary planform. The model assumes incompressible, inviscid, irrotational flow. The program computes forces acting on the wing by modeling the flow with a set of horse shoe vortex elements. It models the flow over an arbitrary wing using two solutions. One solution is the ideal lift, associated with a cambered and twisted wing. The other solution is the additional lift associated with a flat wing. The program computes wing camber and twist using an elliptic loading distribution. The thesis includes the FORTRAN source code, a separable User's Manual for the VORTEX program, discussion of the theory applied in the model, and instructions for operating the program. It shows a sample wing planform with tabular and graphic results.

The thesis also discusses two other CFD models, based on circulation (Γ) and pressure difference (ΔC_p). It presents some of the problems and solutions in grid generation.

TABLE OF CONTENTS

I. INTRODUCTION, PURPOSE AND GOALS

AE2035 is an introductory course in aerodynamics for aeronautical engineering students at Naval Postgraduate School, Monterey, CA. The course must provide students with the required knowledge for future courses in the Aeronautical Engineering Curriculum. The student must understand the techniques used in computational fluid dynamics (CFD) to satisfy that requirement. Powerful computational models are available for the working aerodynamicist, but they are seldom usable in an introductory course. The programming language for these models is usually optimized for computational efficiency rather than teaching, and the documentation may be poor or unavailable. Students need a computer program with graphic output which is less complicated to use as a teaching tool. Providing that computer program is the goal of the thesis.

The program runs on a micro or desk top computer, in keeping with the emphasis on use of individual workstations to supplement the school's mainframe computer. The micro computer has better graphic output programs available, without the extensive programming required on the school mainframe. This flexibility allows the student to modify the output to suit his/her requirements.

The thesis contains five principal sections The first section, **TECHNIQUES**, provides a brief synopsis of three different flow models. The section also provides some insight to certain problems that arise in CFD.

The second section, **VORTEX PROGRAM DEVELOPMENT**, contains a detailed discussion of the horse shoe vortex model. The section includes the theory and techniques used in the VORTEX program.

The third section, **CIRCULATION MODEL**, contains discussion of a circulation (Γ) model. The section includes the theory behind the model and some of the problems encountered.

The fourth section, **PRESSURE DIFFERENCE MODEL**, contains discussion of the pressure difference (ΔC_p) model. The section also includes the theory behind the model and problems encountered.

The fifth and last principal section, the **USERS MANUAL**, is separable from the body of the thesis and provides detailed theory on the horse shoe vortex model. The section also includes instruction on use of the program and an example wing planform with the resulting data.

The computer listing for the VORTEX program is included in the Appendix.

A. PROGRAM ASSUMPTIONS AND REQUIREMENTS

To keep the model streamlined, the following assumptions were made and requirements established.

1. Flow is steady, inviscid, irrotational, and incompressible.
2. The wing surface has zero thickness.
3. The planform has any sweep, taper, and aspect ratio.
4. The program must support low aspect ratio planforms (less than 1.0).
5. The program must contain comments, for easy modification and change.
6. The program results must be available in tabular and graphic form.
7. The program must use accepted variable names. (aspect ratio, taper ratio, etc.)
8. The program must have two options. One is to generate loading for a flat wing shape. The other is to generate a wing shape for an elliptic loading.
9. The programming language must be FORTRAN, with executable and source files provided.

II. TECHNIQUES

Three different models were investigated. Each of the models uses a governing equation for the induced velocity at each control point[1] on the wing. That velocity is associated with the vortex strengths located at field points on tne wing. The induced velocity is a function of two factors. One is the vortex strength and the other is the physical orientation of the field and control points. Flow must be tangent to the wing surface and is found by adding the induced velocity vector (w) to the remote velocity vector (V_∞). The end result is a direct correlation between the wing's shape[2] and the strengths of the vortices. This relationship allows either the strengths of the vortices or the wing shape to be the independent variables.

[1] Field points are the positions of the vortex elements, and control points are the positions where the velocity is evaluated. For example, a control point is where flow tangency is enforced.

[2] Wing shape will be used throughout to describe the surface slope of the wing. For example, one wing shape might be a flat plate, another might have the tip twisted relative to the root chord.

The three models use circulation (Γ), incremental circulation ($\Delta\Gamma$), and pressure difference coefficient (ΔC_p). These quantities are related as follows.

$$\Delta C_p = \frac{2\left(\frac{\Delta\Gamma}{\Delta x}\right)}{V_\infty} = \frac{2\left(\frac{\partial\Gamma}{\partial x}\right)}{V_\infty} \qquad 2.1$$

A. HORSE SHOE VORTEX MODEL

This program models the flow by a set of horse shoe vortices distributed over the wing. Each vortex consists of a bound portion perpendicular to the remote velocity, plus two trailing portions. Each vortex element generates an incremental circulation ($\Delta\Gamma$) around the element.

The program generates a matrix of influence coefficients from an equation for the induced velocity. The program finds the strength of each bound vortex and therefore the incremental circulation around the element by solving the set of simultaneous equations for wing shape. The program can also invert the problem and solve for the wing shape from a specified vortex distribution.

The program can then find the forces and moments on the wing, knowing the distribution of incremental circulation. This solution is possible by using the Kutta-Joukowski theorem which relates incremental circulation and effective velocity to the force generated.

B. CIRCULATION MODEL

This program models the flow by a series of circulation elements distributed over the wing. There is also a circulation sheet that extends from the trailing edge to infinity.

Solution of a set of governing equations provides the strength of each circulation element. The equations satisfy flow tangency on the wing surface and two boundary conditions. Firstly, circulation strength is zero along the leading edge and wing tips. Secondly, the derivative of circulation with respect to chord-wise direction is zero at the trailing edge. The derivative of circulation with respect to chord is vorticity. This trailing edge boundary condition satisfies the Kutta condition of zero vortex strength at the trailing edge. Zero vorticity at the trailing edge also means that the circulation strength behind the wing is a function of the span-wise coordinate only.

C. PRESSURE DISTRIBUTION MODEL

The final model of the flow uses a series of pressure difference elements, ΔC_p, distributed over the wing. The program builds a set of N equations in N unknowns from a governing equation and from the requirement for flow tangency. The boundary conditions are:

1. ΔC_p is zero at the trailing edge, and along the wing tips.

2. For a flat wing, the strength of ΔC_p along the leading edge is unrestrained.

3. For a wing of elliptic lift distribution, the strength of ΔC_p along the leading edge is zero.

D. PROBLEMS INHERENT TO ALL MODELS

1. Field and Control Point Placement

The placement of field and control points within elements is important for all models. Coincident field and control points exhibit singularities. Each model uses different methods to handle them. Direct integration eliminates the singularity in two models. The other requires a special arrangement of control and field points along the leading edge and wing tips.

2. Grid Size and Solution Speed

Fine grids require more time for solution and larger computers. All models compromise between data quantity and speed. The circulation model is very memory intensive and will only run on the mainframe computer. The other models will run on a desktop computer, though the time required increases rapidly with finer grids.

3. Grid Element Shapes

Finite elements in a non-rectangular planform can be either trapezoids, or rectangles. Trapezoids match the planform exactly; rectangles present a ragged leading and/or trailing edge. The form to use is dependent on the type of model. The circulation model can use a trapezoid without difficulty. Coordinate transformations easily handle the shape and at the same time allow non-uniform cosine spacing of the elements. The Vortex and ΔC_p models work best with rectangular grid elements. Therefore the planform is not an exact representation of the wing. This modification is reasonable since the forces and moments computed are comparable to other models.

E. SCOPE OF DISCUSSION

Of the three models investigated, only the VORTEX program was completely successful. Significant effort was expended on the CIRCULATION and ΔC_p models trying to make them functional. Rather than expend a large portion of this thesis discussing all aspects of the two unsuccessful models, an overview of them will be presented and the relevant factors of each will be addressed. Since the VORTEX program was successful, and is completely functional, a detailed discussion of the procedure and assumptions is provided. In keeping with this, the FORTRAN source code for the VORTEX model is the only computer code included.

III. VORTEX PROGRAM DEVELOPMENT

A. BASIS OF THE VORTEX MODEL

The VORTEX program is an adaptation of a program by Moran [Ref.1]. The program models the flow by a series of horse shoe vortices distributed over the wing. In his text, Moran [Ref.1] develops a simple program to find the strengths of a series of horse shoe vortices. His program is rather limited in that it only works for straight wings without taper or sweep. Additionally, his program only finds the loading generated by a flat wing. The final limitation is his use of uniform sized grid elements, which fails to concentrate grid elements near the boundaries of the wing.

As with many aerodynamic problems, the VORTEX program uses a grid or mesh of finite sized elements distributed over the surface of the wing. The program divides the wing planform (there is zero thickness modeled in this program) into N discrete elements.

The program solves a set of N equations for N unknowns. The equations are derived from an induced velocity equation and the requirement for flow tangency on each grid element. The N unknowns are the strengths of the individual horse shoe vortices associated with each element.

Knowing the strengths of the vortex elements, the program can find the forces and moments acting on the wing as a result of those elements.

B. MODIFICATIONS TO THE MODEL

The VORLAT program, developed by Moran [Ref.1], required changes to suit the goals of the thesis. Items requiring improvement were the ability to handle sweep and taper. Other improvements were also added to generate the grid using cosine spacing. A module to determine camber and twist, called shape, was also incorporated.

C. TECHNIQUE FOR A FLAT WING

This is a short description of the VORTEX program. Consult Moran [Ref.1] for a description of the VORLAT program, the foundation of the VORTEX program.

1. Down-wash and the Relationship to Flow Tangency

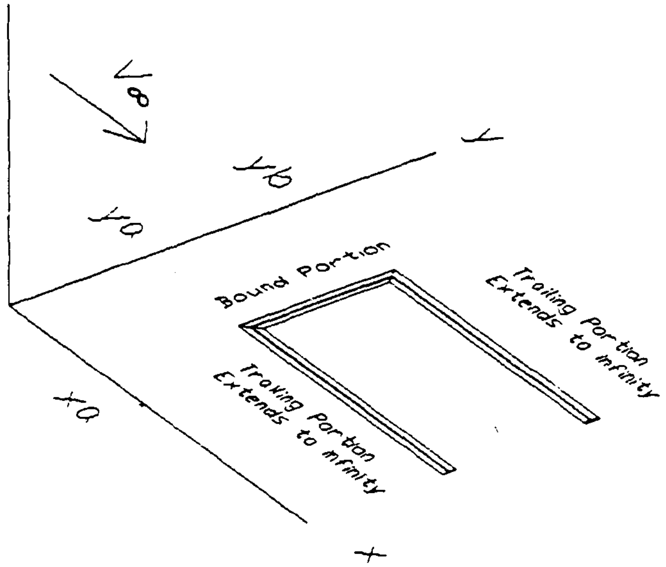
A horse shoe vortex is located in the xy plane as shown in Figure 3-1. The vortex induces a velocity at any point in the xy plane. That velocity can be determined from equation 3.1, as follows:

$$w_{ij}(x,y) = \frac{\Delta\Gamma}{4\pi(y-y_a)}\left\{1+\frac{\sqrt{(x-x_a)^2+(y-y_a)^2}}{(x-x_a)}\right\} - \frac{\Delta\Gamma}{4\pi(y-y_b)}\left\{1+\frac{\sqrt{(x-x_a)^2+(y-y_b)^2}}{(x-x_a)}\right\} \quad 3.1$$

If $x = x_a$ then,

$$w_{ij}(x,y) = \frac{\Delta\Gamma}{4\pi}\left(\frac{1}{y-y_a} - \frac{1}{y-y_b}\right) \qquad 3.2$$

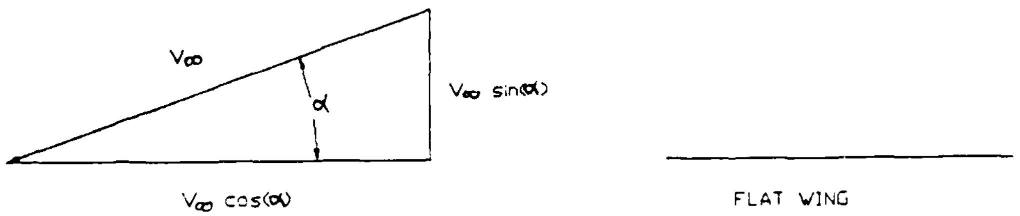
$w_{ij}(x,y)$ is the velocity induced at the control point (x,y). $\Delta\Gamma$ is the incremental circulation[1], and (x_a,y_a) and (x_a,y_b) are the coordinates of the corners of the horse shoe vortex. Moran derived this equation using the Biot-Savart Law [Ref.1].



A Horse Shoe Vortex in the Wing Plane
Figure 3-1

[1] Different authors use different variable names for Circulation and Vorticity. In the Vortex model, $\Delta\Gamma$ is used for incremental circulation. In the Circulation model, Γ is used for Circulation strength. Caution is urged when relating one model to another.

As shown in Figure 3-2, the remote velocity, with a wing at angle of attack α, has components $V_\infty cos(\alpha)$ and $V_\infty sin(\alpha)$.



Velocity Components on a Flat Wing
Figure 3-2

If the flow is tangent to the wing, the down-wash velocity[2] at a point on the wing must equal $V_\infty sin(\alpha)$. Each vortex on the wing contributes to the down-wash[3] at every point on the wing. So, an equation can be developed for each control point as a function of the strength of each horse shoe vortex. In this thesis, the mid point of any bound vortex is termed a field point. Any point on the wing where flow tangency is evaluated is termed a control point.

[2] Downwash is considered positive when its direction is downward.

[3] The contribution will be positive when the induced velocity is downward, or negative when the induced velocity is upward.

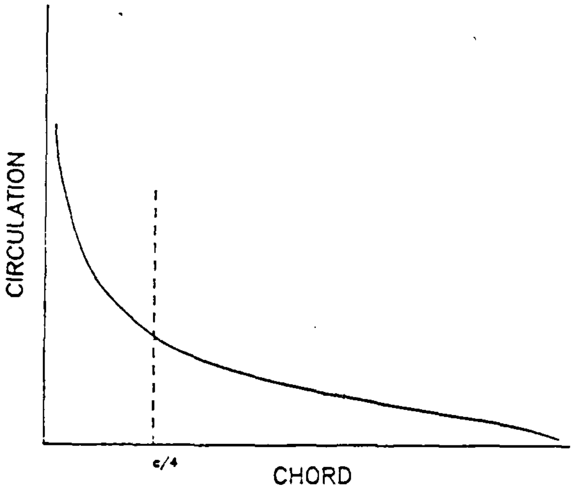
An example of the equation for the control point (x_p, y_p) is.

$$V_\infty \sin\alpha - \sum_{ij} \Gamma_{ij} w_{ij}(x_p, y_p) = 0 \qquad 3.3$$

Combining all the control points results in a linear set of N equations in N unknowns.

2. Placement of the Horse Shoe Vortex

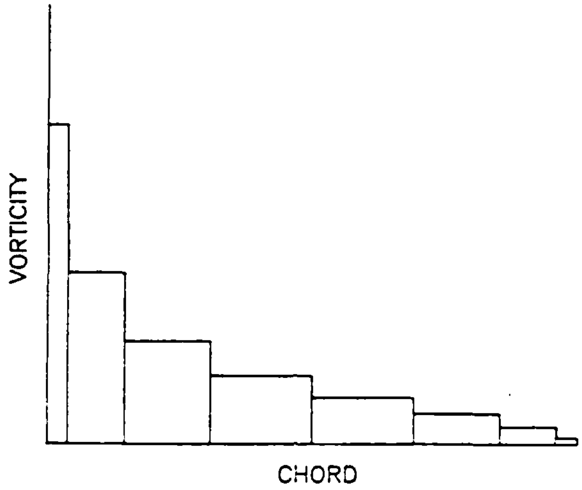
Moran [Ref.1] and others have discussed the placement of a lumped vortex on a two-dimensional airfoil with one chord-wise element. For the two-dimensional case, the vortex is placed at the quarter chord point and flow tangency is evaluated at the three-quarter chord point. The two-dimensional reasoning can be applied to a wing of finite span, assuming the wing is formed with a single chord-wise element.



Circulation Distribution over a Flat Airfoil
Figure 3-3

For the flat airfoil, the circulation distribution is known to be approximately as shown in Figure 3-3. The centroid of this circulation distribution is at the quarter chord point. As a result, it is reasonable to lump the total circulation at the quarter chord point. For these conditions, the flow is tangent at the three quarter chord point. This arrangement correctly computes the moment coefficient (C_m) for the two dimensional flat airfoil.

The reason for other authors' placement of the vortex at the quarter chord of each grid element was questioned during development of the VORTEX program. The other authors used lifting line theory with a single chord-wise element for their quarter chord vortex placement. The VORTEX program does not use a single chord-wise element. The wing is divided into a number of individual elements which are modeled with a uniform distribution of vorticity over each element. The centroid is at the center of the element. So, for the arrangement in Figure 3-4, the incremental circulation is concentrated at the center of each element.



Approximation of Vorticity with Individual Elements
Figure 3-4

With the circulation concentrated at the center of each element, the flow tangency point is set at one half an element downstream of the concentrated circulation and falls at the border between elements, or at the trailing edge of the wing.

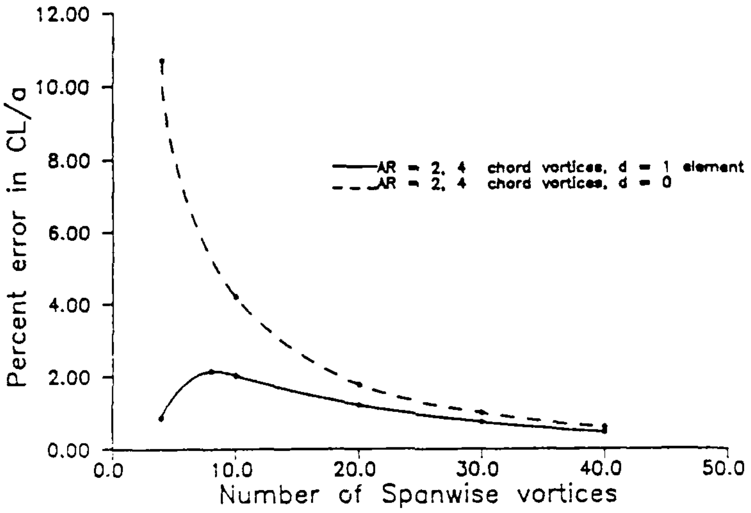
The coefficients affected by the vortex placement are C_M and X_{cp}. In Ref.2, X_{cp}/c for a straight flat wing, aspect ratio 2, was determined to be 0.209. With the lumped vortex at the mid chord point, the location of X_{cp}/c converged to a value of approximately 0.234. The error indicates the vortex is too far aft on the element. Relocating the lumped vortex element to the quarter chord point, and evaluating flow tangency at the three-quarter chord point, moved the computed location of X_{cp}/c forward, but still not to the true value of 0.209.

In Ref.3, Hough looked at optimum grid and vortex arrangement for rapid convergence. One of the planforms was a straight flat wing with an aspect ratio of 2. Hough used a conventional vortex lattice arrangement of uniform size grid elements with the lumped vortex at the quarter chord point. For the aspect ratio 2 wing, he found that the computed C_L/α was larger than actual and that the value converged as in Figure 3-5. It was expected that the error in C_L/α could be reduced by decreasing the planform area by some factor. He found that convergence of C_L/α, Vortex Drag Factor (K)[4], and X_{cp} was improved by insetting the tip vortex by some fraction (d) of a grid element. This inset improved convergence dramatically when d = 1/4. Though not addressed by Moran [Ref.1], that is the reason for insetting the grid by d = 1/4.

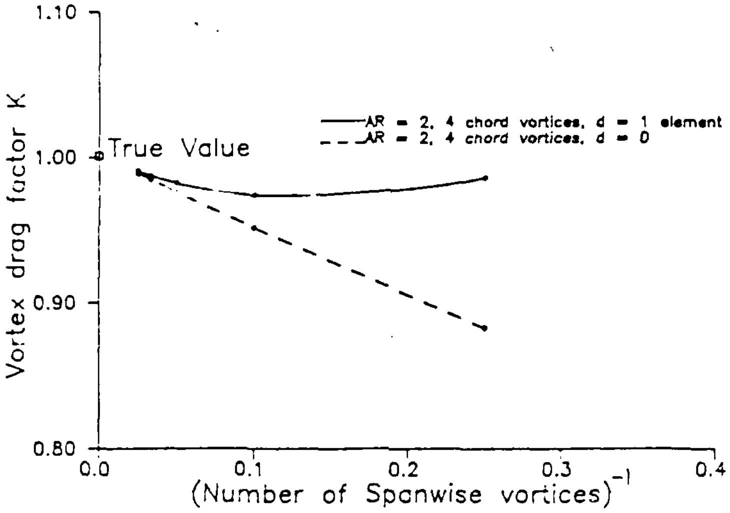
The VORTEX program uses cosine spacing instead of uniform spacing so insetting the grid by d = 1/4 was not an available option. Instead, the span-wise grid layout is developed with an additional row of tip elements which are not used in the computation of incremental circulation or force. This reduces the planform area and the improvement in convergence can be seen in Figures 3-5 and 3-6.

[4] Vortex Drag Factor, K is; $\frac{\pi \cdot AR \cdot C_{Di}}{C_L^2}$

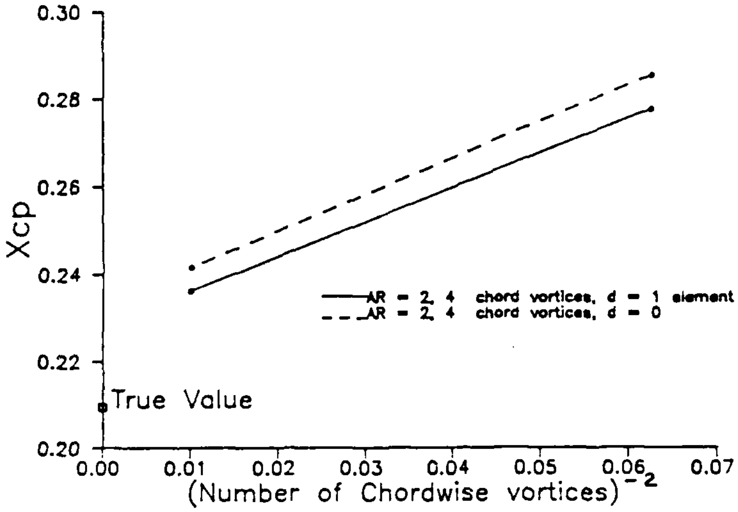
Additionally, the convergence of X_{cp} is improved, as seen in Figure 3-7.



Percent Error in CL/a
Figure 3-5



Vortex Drag Factor (K)
Figure 3-6



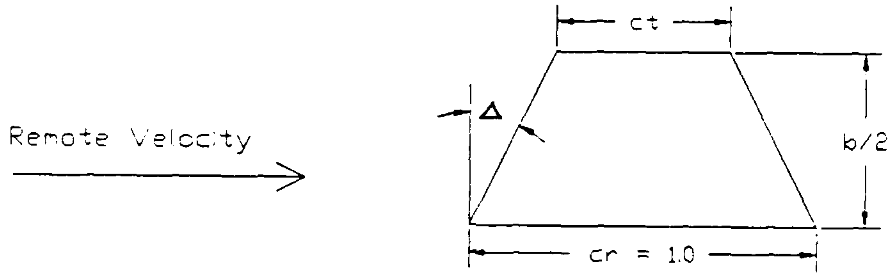
X_{cp}
Figure 3-7

3. Satisfaction of the Kutta Condition

To satisfy the Kutta condition, the vortex strength must be zero at the trailing edge. While not explicitly required by the VORTEX program, the resultant vortex strengths, for a flat wing, approach zero at the trailing edge. Therefore, the results seem to imply that flow tangency at the trailing edge of a flat wing is a corollary of the Kutta condition.

4. Planform Development

It is necessary to develop wing geometry from aspect ratio, taper ratio and sweep angle. Aspect ratio is the span divided by the average chord. Taper ratio is the tip chord divided by the root chord, and sweep angle is the angle between the leading edge and the y axis. The y axis is perpendicular to the remote velocity and parallel to the earth. The aircraft design courses taught at NPS use these variables to define the planform. *Moran* [Ref.1] uses a fixed root chord equal to 1.0 and varies the span to achieve different aspect ratios. The VORTEX program adopts this approach, which works well.

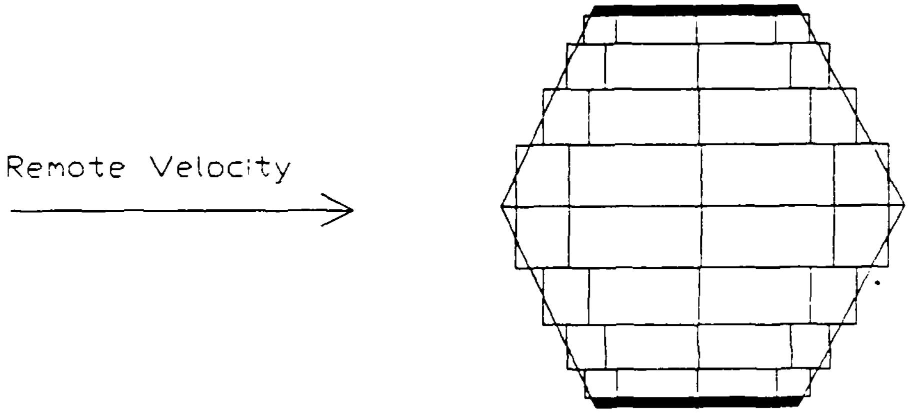


Planform and Variables used in VORTEX Program
Figure 3-8

The subroutine SET85 determines planform variables and stores them in part of the matrices WING and SECTN for further use. The WING and SECTN matrices also contain final results of the program.

5. Grid Development

The program develops a grid to model the vortex system after establishing the outline of the planform. The grid elements are rectangular. The number of elements must be small to keep the time required for solution reasonable. The program concentrates grid elements in areas where the rate of change is most rapid, or where the values are most important. Those areas are along the boundaries. An excellent method for concentrating the points in the areas desired while minimizing the total number of points is through cosine spacing. The program uses a cosine function to distribute span-wise and chord-wise grid points in a non-uniform fashion after a suitable coordinate transformation. Figure 3-9 contains a sample of the layout. Notice how the rectangular elements model the planform.



Grid Model with Cosine Spacing
Figure 3-9

6. Solving the Set of Equations

The method used to solve the set of N linear equations is arbitrary. Moran [Ref. 1] uses a Gaussian algorithm for solution of an N by N matrix. The VORTEX program uses the same algorithm. Increasing the total number of grid elements beyond 100 would require modification of the GAUSS subroutine.

7. Finding the Forces on the Elements

The Kutta-Joukowski theorem states that the force per unit span acting on an element is.

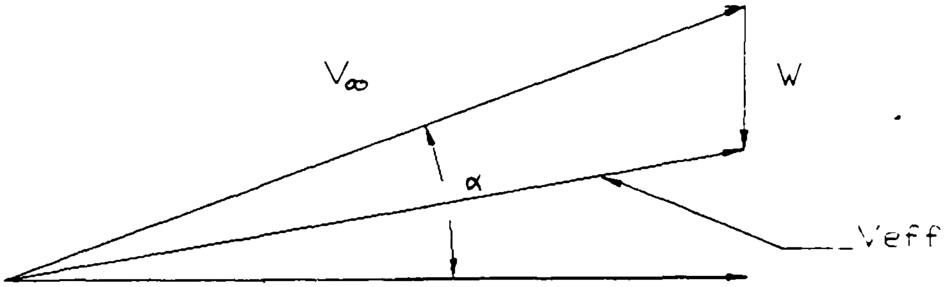
$$\frac{\overrightarrow{\Delta F}}{\Delta y} = \rho \, \overrightarrow{V_{eff}} \, x \, \overrightarrow{\Delta \Gamma} \qquad 3.4$$

In this equation, $\overline{V_{eff}}$ is the local effective velocity at the center of the element. $\overline{V_{eff}}$ is defined in equation 3.5 and Figure 3-10. ρ is the density and $\Delta\Gamma$ is the incremental circulation around the element. This circulation ($\Delta\Gamma$) is the same $\Delta\Gamma$ contained in equation 3.1, and can be termed the incremental circulation that occurs over the element. Throughout this section, certain approximations and dimensional simplifications will be made. The first.is the small angle approximation, where the sine is approximately equal to the angle in radians, and the cosine is approximately equal to 1.0. This approximation requires that angle of attack for the flat wing be small, generally less than 10 degrees. The approximation also requires that any wing slope on the elliptically loaded wing be small. This requirement is met by restricting the desired lift coefficient to values less than about 0.5. In the process of determining coefficients of lift, drag and moment, certain dimensional reference quantities arise. These quantities are density (ρ), remote velocity (V_∞), planform area (S), and average chord ($\bar{c}$) . In performing dimensional analysis an arbitrarily value of unity can be assigned to these terms.

a. Finding ΔC_p

The distribution of ΔC_p over the wing is desired, so the force on each element must be converted to a dimensionless pressure difference coefficient. $\overline{V_{eff}}$ is defined relative to the wing as;

$$\overrightarrow{V_{eff}} = V_{\infty}\cos\alpha\,\vec{i} + \left(V_{\infty}\sin\alpha - w\right)\vec{j} \qquad 3.5$$



Components of Effective Velocity ($\overline{V_{eff}}$)
Figure 3-10

It is also significant to note that $\Delta\Gamma$ can be written as;

$$\overrightarrow{\Delta\Gamma} = \Delta\Gamma\,\vec{j} \qquad 3.6$$

since the bound portion of the horse shoe vortex is aligned with the y axis. The result of the force cross product, Equation 3.4, is;

$$\frac{\overrightarrow{\Delta r}}{\Delta y} = \rho\left|\left(w - V_{\infty} sin\, a\right)\Delta\Gamma\vec{i} + V_{\infty} cos\, a\, \Delta\Gamma\vec{k}\right| \qquad 3.7$$

Setting $\rho = 1$, $V_{\infty} = 1$, and incorporating the small angle approximation;

$$\frac{\overrightarrow{\Delta F}}{\Delta y} = \left|\left(w - a\right)\Delta\Gamma\vec{i} + \Delta\Gamma\vec{k}\right| \qquad 3.8$$

This force is related to ΔC_p. ΔC_p is a scalar, rather than a vector quantity, so the program uses the force component normal to the wing to compute ΔC_p. That component is;

$$\Delta F = \Delta y \Delta\Gamma \qquad 3.9$$

Making the force non-dimensional gives a pressure difference coefficient, ΔC_p.

$$\Delta C_p = \frac{\Delta F}{\frac{1}{2}\rho V_{\infty}^2 \Delta x \Delta y} \qquad 3.10$$

Finally, after setting ρ - 1 and V_∞ - 1, ΔC_p for an element is;

$$\Delta C_p = \frac{2\,\Delta F}{\Delta x \Delta y} = \frac{2\,\Delta\Gamma}{\Delta x} \qquad 3.11$$

This equation provides the ΔC_p for each element, which can be plotted versus chord, for each span-wise section.

b. Finding the lift, drag and moment

Lift, drag and moment are also found from the vector force on each element. Equation 3.7 showed that the force on an element has components normal and tangent to the wing. Once more setting ρ - 1, and V_∞ - 1, the vector force on an element is;

$$\overrightarrow{\Delta F} = \left|\left(w - \sin\alpha\right)\Delta\Gamma\Delta y\,\vec{i} + \cos\alpha\,\Delta\Gamma\Delta y\,\vec{k}\right| \qquad 3.12$$

These forces are oriented relative to the wing coordinate system, where $\vec{i}$ is parallel to the wing and $\vec{k}$ is perpendicular to the wing. Lift and drag are normal and tangent, respectively, to the remote velocity (V_∞). The wing is at an angle of attack (α) to that remote velocity. Using this information, the program transforms the forces to a coordinate system oriented to the remote velocity. For an individual element the lift and drag forces are;

$$\Delta L = Elemental\,Lift\,Force = (\cos\alpha)(\Delta y\Delta\Gamma)(\cos\alpha) - (w - \sin\alpha)(\Delta y\Delta\Gamma)(\sin\alpha)$$

$$\Delta D = Elemental\,Drag\,Force = (\cos\alpha)(\Delta y\Delta\Gamma)(\sin\alpha) + (w - \sin\alpha)(\Delta y\Delta\Gamma)(\cos\alpha)$$

using a small angle approximation, and discarding higher order terms, 3.13

$$\Delta L = Elemental\,Lift\,Force = \Delta y\Delta\Gamma - w\alpha\Delta y\Delta\Gamma = \Delta y\Delta\Gamma$$

$$\Delta D = Elemental\,Drag\,Force = \Delta y\Delta\Gamma\alpha + (w-\alpha)\Delta y\Delta\Gamma = w\Delta y\Delta\Gamma$$

The total lift and total drag acting on the wing is a summation of the elemental lifts and drags. Using wing symmetry;

$$L = Total\,Lift = 2\sum_{semi-span}\Delta L \qquad 3.14$$

$$D = Total\,Drag = 2\sum_{semi-span}\Delta D$$

The moment about the y axis for an individual element, with the accepted sign convention, is;

$$\Delta M = Elemental\,Moment = -(normal\,wing\,force)\,(x\,position\,of\,center\,of\,element) \qquad 3.15$$

$$\Delta M = Elemental\,Moment = (-\Delta y\Delta\Gamma)(x_{cen})$$

Total moment about the y axis is a summation of the elemental moments. Once again, using wing symmetry.

$$M = Total\,Moment = 2 \sum_{semi-span} \Delta M \qquad 3.16$$

8. Finding the Coefficients

Non-dimensionalizing the total lift and drag gives,

$$C_L = \frac{L}{\frac{1}{2}\rho V_\infty^2 S}$$

$$C_D = \frac{D}{\frac{1}{2}\rho V_\infty^2 S} \qquad 3.17$$

where S = span x average chord = (b x $\bar{c}$), ρ = 1 and V_∞ = 1.

Non-dimensionalizing the total moment about' the y axis gives,

$$C_{M_o} = \frac{M}{\frac{1}{2}\rho V_\infty^2 \bar{c} S} \qquad 3.18$$

where $\bar{c}$ is the average chord.

The aerodynamic center X_{ac} is important. X_{ac} is the chord-wise position about which the flat wing has zero moment, and is found from.

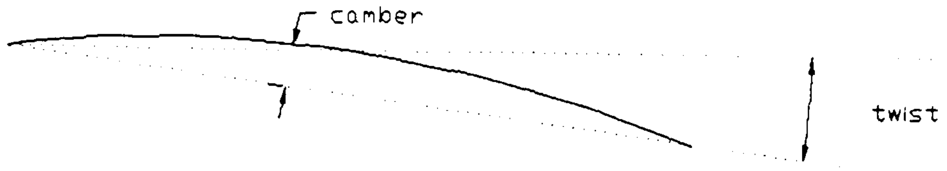
$$X_{ac} = \frac{-\,total\,moment}{total\,lift} \qquad 3.19$$

This equation assumes that total moment is taken about the y axis, or x=0. When the program finds the moment on the elliptically loaded wing, it uses the X_{ac} from the flat wing as the reference axis and finds C_{Mac}.

D. TECHNIQUE FOR SPECIFIED LOADING

1. Finding Camber and Twist from Loading

Elliptic loading in a span-wise direction results in minimum induced drag, and loading is proportional to circulation. At sufficiently high aspect ratio, a flat wing with elliptic area distribution generates elliptic loading but weighs more than a straight wing. Wings with straight leading and trailing edges also cost less to manufacture than elliptic wings. So, the program generates a span-wise elliptic lift distribution by slightly twisting the wing. The direct relationship between wing shape (camber and twist) and circulation distribution makes this generation possible.



Camber and Twist on the Elliptically Loaded Wing
Figure 3-10

For a thin wing, elliptic load distribution in the chord-wise direction requires an approximately parabolic camber shape. The program specifies an elliptic chord-wise load distribution and then finds the wings' associated shape.

2. Specifying the Lift Coefficient

Specifying the form and amplitude of the ideal lift distribution over the wing determines the form and amplitude of the wing shape. A load distribution of elliptic **form** is imposed in this case and the corresponding **amplitude** is fixed by the desired ideal wing lift coefficient, C_{Li}.

3. Maximum Desired Lift Coefficient

The small angle approximation breaks down when large lift coefficients are specified. To prevent this occurrence, the maximum desired lift coefficient is restricted to 0.5.

4. Achieving the Desired Lift Coefficient

The program scales a reference distribution of ΔC_p, which is elliptic in form, to achieve the desired total lift coefficient for the wing.

5. Forces, Moments and Coefficients

Once the ΔC_p distribution is scaled for the elliptically loaded wing, the program finds forces, moments and coefficients in the same manner as for the flat wing. The only difference is in the coordinate transformation used to convert forces on the wing to lift and drag. The individual elements on the elliptically loaded wing are no longer at a uniform angle to the remote velocity. This variation in angle must be taken into account when performing the coordinate transformation which gives lift and drag forces on individual elements.

The program finds moment coefficient about the aerodynamic center, C_{Mac}, from,

$$C_{Mac} = C_{Mo} + C_{Li}\left(\frac{x_{ac}}{\bar{c}}\right) \qquad 3.20$$

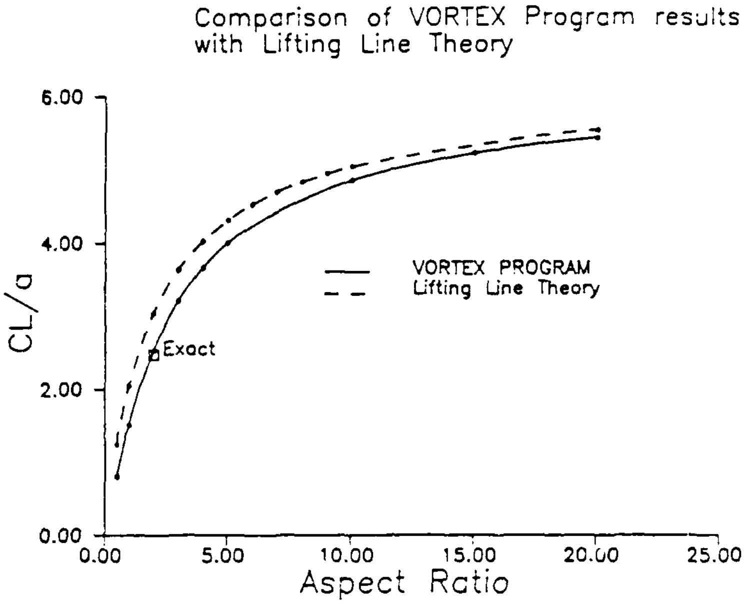
where X_{ac} is found from the flat wing, and $\bar{c}$ is the average chord. C_{Mo} is the pitching moment of the elliptically loaded wing about the y axis, and C_{Li} is the lift coefficient of the elliptically loaded wing.

E. VALIDATION OF RESULTS

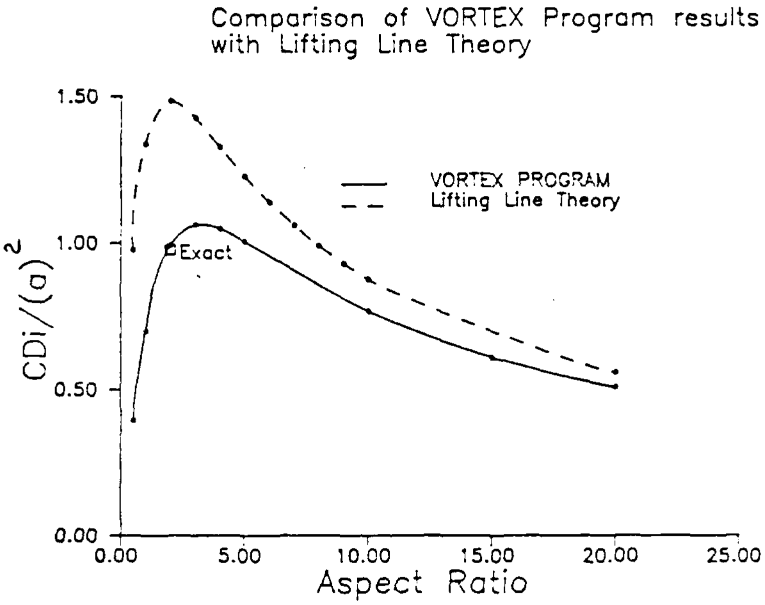
Two sources are used to check the validity of the VORTEX program results. Lifting line theory provides one source of predicted lift and drag values for straight high aspect ratio wings A continuous loading method developed by the National Aerospace Laboratory of the Netherlands (NLR), presented in Ref.2, is used as the second source. The method used by NLR is a very accurate computational method and is considered to be exact for this comparison.

Kuethe and Chow [Ref.4] discuss lifting line theory for the case of a flat, untapered, rectangular wing. Using lifting line theory and the VORTEX program, lift and drag coefficients for identical wings were computed. A range of aspect rations from 0.5 through 20 were selected. Lifting line theory is known to be reliable at higher aspect ratios, but tends to lose accuracy at low aspect ratios.

As the graphic results in Figures 3-11 and 3-12 demonstrate, CL/α and $CDi/(\alpha)^2$ values obtained from lifting line theory and the VORTEX program converge nicely at high aspect ratios. At lower aspect ratio, lifting line and the VORTEX program show a difference in calculated value. The values of CL/α and $CDi/(\alpha)^2$ obtained by NLR [Ref.2] agree closely with the VORTEX program results at aspect ratio 2. The NLR results validate the VORTEX program results for straight wings. Exact data from swept and tapered wings were not investigated.



Comparison of Lift
Figure 3-11



Comparison of Drag
Figure 3-12

IV. CIRCULATION MODEL

A. BASIS OF THE CIRCULATION MODEL

This program models the flow with a sheet of distributed circulation and a continuous trailing vortex sheet behind the wing. Barna [Ref.5], and also Milne-Thomson [Ref.6, pages 171-177], discuss this conceptually simple model. The program uses the wing shape, flow tangency, boundary conditions, and the Biot-Savart Law to develop a set of equations. The program then solves the set of equations for the unknown circulation strengths at all grid points on the wing.

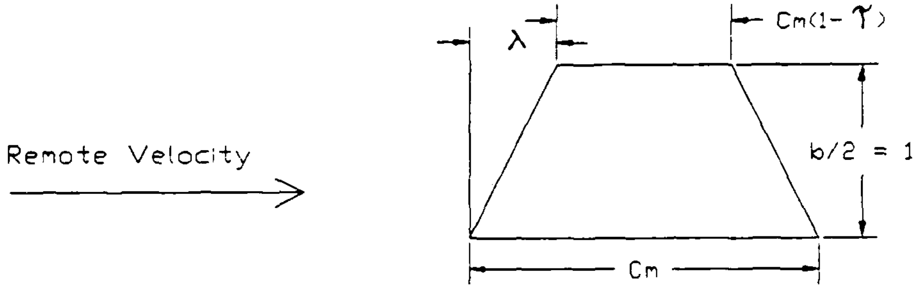
B. TECHNIQUE FOR SOLUTION OF THE CIRCULATION MODEL

1. Induced Velocity

The velocity induced at a point by the distributed circulation sheet on the wing and the trailing vortex sheet can be treated as the sum of the velocity induced by each. The velocity induced at the control point (x_o, y_o) by field point (x,y), located on the wing is.

$$w\Big|_{wing} = \int_{-1}^{+1} \left\{ \frac{1}{V_\infty 4\pi} \int_{x_L}^{x_T} \frac{1}{r^3} \left| (x-x_o)\left(\frac{\partial\Gamma}{\partial x}\right) + (y-y_o)\left(\frac{\partial\Gamma}{\partial y}\right) \right| dx \right\} dy \qquad 4.1$$

r is the distance between the field and control points, and Γ is the strength of the circulation at the field point (x,y). Setting the semi-span length equal to unity, and varying the chord accommodates different aspect and taper ratios.



Planform and Variables used in Circulation Model
Figure 4-1

Circulation (Γ), and pressure difference coefficient (ΔC_p) are related as follows;

$$\Delta C_p = \frac{2\left(\frac{\partial \Gamma}{\partial x}\right)}{V_\infty} \qquad 4.2$$

In equation 4.2, ΔC_p is the pressure difference coefficient between the upper and lower surfaces, and Γ is the circulation.

The velocity induced at control point (x_o, y_o) by the segment of the trailing vortex sheet attached to the wing at (x_t, y), is:

$$w\Big|_{trail} = \int_{-1}^{+1} \frac{1}{V_\infty 4\pi r}\left\{\frac{r-(x_T-x_o)}{(y-y_o)}\right\}\left(\frac{\partial\Gamma_T}{\partial y}\right)dy \qquad 4.3$$

r is the distance between the points, and Γ_t is the strength of the circulation at the field point (x_t, y) on the trailing edge.

When the remote velocity is unity, the induced velocity is the same as the slope of the wing at the point in question, and the total slope is:

$$\left(\frac{\partial z}{\partial x_o}\right) = \left(\frac{\partial z}{\partial x_o}\right)_{wing} + \left(\frac{\partial z}{\partial x_o}\right)_{trail} \qquad 4.4$$

additionally,

$$\left(\frac{\partial z}{\partial x_o}\right)_{wing} = \frac{w\Big|_{wing}}{V_\infty} \qquad 4.5$$

and

$$\left(\frac{\partial z}{\partial x_o}\right)_{trail} = \frac{w\Big|_{trail}}{V_\infty} \qquad 4.6$$

The program uses these equations to satisfy flow tangency on the wing with a known shape and solves for the unknown circulation strengths. In matrix notation the equations have the following form.

$$\left[A \right] \left\{ \frac{\partial \Gamma}{\partial x} \right\} + \left[B \right] \left\{ \frac{\partial \Gamma}{\partial y} \right\} = \left\{ \frac{\partial z}{\partial x_o} \right\} \qquad 4.7$$

2. Boundary Conditions

The requirement for flow tangency at each element on the wing is not enough to find the circulation which satisfies the Kutta condition; the program must also enforce certain boundary conditions.

The flat wing, which produces the additional lift, has the following boundary conditions:

1. $\Gamma = 0$ along the leading edge and the wing tips.
2. $\partial\Gamma/\partial x = 0$ along the trailing edge.

The elliptically loaded wing, which produces the ideal lift, has the following boundary conditions:

1. $\Gamma = 0$ along the leading edge, and wing tips.
2. $\partial\Gamma/\partial x = 0$ along both the leading and trailing edges.

Using a finite difference scheme, it is possible to approximate the partial derivatives of Γ ($\partial\Gamma/\partial x$, & $\partial\Gamma/\partial y$) from the values of Γ at neighboring elements and satisfy these boundary conditions. The partial derivative of Γ with respect to x, namely ($\partial\Gamma/\partial x$), approaches infinity at the leading edge of the flat wing. This singularity makes finite differencing near the leading edge difficult. Cosine spacing is used to minimize that problem and also to spacing is used to minimize that problem and also to concentrate elements near the edges of the wing. The x coordinate in the chord-wise direction can be expressed in an alternate form by ϕ, where:

$$\phi = cos^{-1}\left\{1 - \frac{2(x - \lambda\sigma y)}{c}\right\} \qquad 4.8$$

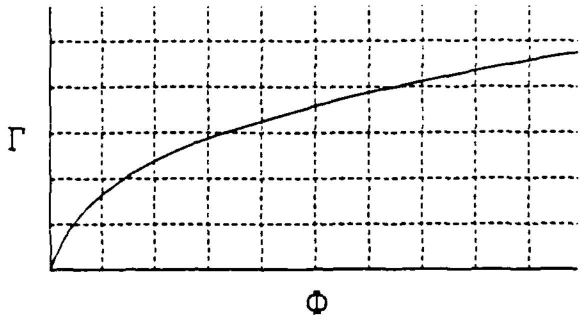
The additional variables are defined in equation 4.24 and Figure 4-3. The y coordinate in the span-wise direction becomes θ, where:

$$\theta = cos^{-1}(y) \qquad 4.9$$

After this transformation,;

$$\frac{\partial\Gamma}{\partial\phi} = \frac{\partial\Gamma}{\partial x}\left(\frac{c}{2}sin(\phi)\right) \qquad 4.10$$

From the transformation, it is possible to show that $\partial\Gamma/\partial\phi$ has a finite value at the leading edge even when $\partial\Gamma/\partial x$ is infinite.



Circulation vs Phi
Figure 4-2

Using Figure 4-2 as a guide, the program estimates the value of $\partial\Gamma/\partial\phi$ from the values of Γ on neighboring elements. The following shows the procedure for finding $\partial\Gamma/\partial\phi$ at the leading edge element as a function of Γ on neighboring elements. Other points are derived in a similar fashion.

$$\Gamma_\phi = A_1\phi + A_2\phi^2 + A_3\phi^3 \qquad 4.11$$

$$\frac{d\Gamma}{d\phi} = A_1 + 2A_2\phi + 3A_3\phi^2 \qquad 4.12$$

For the first element, $\phi_1 = \frac{\Delta\phi}{2}$

For the second element, $\phi_2 = \frac{3\Delta\phi}{2}$

For the third element, $\phi_3 = \frac{5\Delta\phi}{2}$

$$A_1 \frac{\Delta\phi}{2} + A_2\left(\frac{\Delta\phi}{2}\right)^2 + A_3\left(\frac{\Delta\phi}{2}\right)^3 = \Gamma_{\phi_1} \tag{4.13a}$$

$$A_1 \frac{3\,\Delta\phi}{2} + A_2\left(\frac{3\,\Delta\phi}{2}\right)^2 + A_3\left(\frac{3\,\Delta\phi}{2}\right)^3 = \Gamma_{\phi_2} \tag{4.13b}$$

$$A_1 \frac{5\,\Delta\phi}{2} + A_2\left(\frac{5\,\Delta\phi}{2}\right)^2 + A_3\left(\frac{5\,\Delta\phi}{2}\right)^3 = \Gamma_{\phi_3} \tag{4.13c}$$

which becomes

$$\begin{bmatrix} 0.5 & 0.25 & 0.125 \\ 1.5 & 2.25 & 3.375 \\ 2.5 & 6.25 & 15.625 \end{bmatrix} \begin{Bmatrix} A_1\Delta\phi \\ A_2\left(\Delta\phi\right)^2 \\ A_3\left(\Delta\phi\right)^3 \end{Bmatrix} = \begin{Bmatrix} \Gamma_{\phi_1} \\ \Gamma_{\phi_2} \\ \Gamma_{\phi_3} \end{Bmatrix} \tag{4.14}$$

and

$$A_1\Delta\phi = 3.75\,\Gamma_{\phi_1} - .833\,\Gamma_{\phi_2} + .15\,\Gamma_{\phi_3} \tag{4.15a}$$

$$A_2\Delta\phi^2 = -4.0\,\Gamma_{\phi_1} + 2.0\,\Gamma_{\phi_2} - .4\,\Gamma_{\phi_3} \tag{4.15b}$$

$$A_3\Delta\phi^3 = 1\,\Gamma_{\phi_1} - .6667\,\Gamma_{\phi_2} + .2\,\Gamma_{\phi_3} \tag{4.15c}$$

$$\left(\frac{d\Gamma}{d\phi} \right)_1 = A_1 + 2A_2 \frac{\Delta\phi}{2} + 3A_3 \left(\frac{\Delta\phi}{2} \right)^2 \qquad 4.16$$

when 4.16 is combined with 4.15, it becomes;

$$\left(\frac{d\Gamma}{d\phi} \right)_1 = \frac{1}{\Delta\phi} \left\{ .5\,\Gamma_{\phi_1} + .6667\,\Gamma_{\phi_2} - .1\,\Gamma_{\phi_3} \right\} \qquad 4.17$$

The uncertainty in this approximation is of order $\Delta\phi^4$. For interior elements,

$$\left(\frac{d\Gamma}{d\phi} \right)_i = \frac{1}{\Delta\phi} \left\{ -.5\,\Gamma_{\phi_{i-1}} + .5\,\Gamma_{\phi_{i+1}} \right\} \qquad 4.18$$

and for the trailing edge element,

$$\left(\frac{d\Gamma}{d\phi} \right)_i = \frac{1}{\Delta\phi} \left\{ -.1522\,\Gamma_{\phi_{i-2}} + .9565\,\Gamma_{\phi_{i-1}} - .8043\,\Gamma_{\phi_i} \right\} \qquad 4.19$$

Similar relations hold in the span-wise direction. In matrix form:

$$\left[CC \right] \left\{ \Gamma \right\} = \left\{ \frac{\partial\Gamma}{\partial\phi} \right\} \qquad 4.20$$

$$\left[BB \right] \left\{ \Gamma \right\} = \left\{ \frac{\partial\Gamma}{\partial\theta} \right\} \qquad 4.21$$

3. Sweep and Taper

The program must model swept and tapered wings. It makes a series of coordinate transformations to accomplish this. The final result is an equation for the slope due to the circulation on the wing.

$$\left.\frac{\partial z}{\partial x_o}\right|_{wing} = \frac{1}{4\pi}\int_0^{\pi}\int_0^{\pi}\left|\left(\frac{\mathring{x} - \mathring{x}_o}{r^3}\right)sin\theta\left(\frac{\partial\Gamma}{\partial\phi}\right) - \left(\frac{y - y_o}{r^3}\right)\frac{c\,sin\phi}{2}\left(\frac{\partial\Gamma}{\partial\theta}\right)\right| d\phi\, d\theta \qquad 4.22$$

and for the contribution of the trailing vortex filament,

$$\left.\frac{\partial z}{\partial x_o}\right|_{trail} = \frac{-1}{4\pi}\int_0^{\pi}\frac{1}{r_T}\left|\frac{r_T - (x_T - x_o)}{(y - y_o)}\right|\left(\frac{\partial\Gamma}{\partial\theta}\right) d\theta \qquad 4.23$$

where

$$x^{\circ} = (x - \beta y)$$

$$x^{\circ}_{o} = (x_{o} - \beta y_{o})$$

$$\beta = (\lambda - c_{m} \iota \zeta)$$

$$\lambda = tan(\Delta),\ tangent\ of leading\ edge\ sweep$$

$$c_{m} = root\ chord$$

$$\iota = 1 - taper\ ratio$$

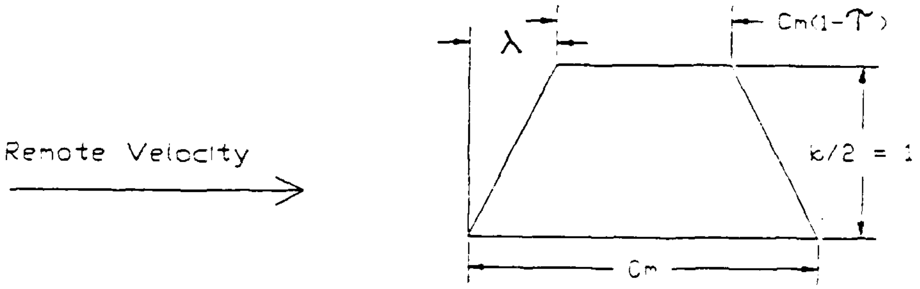
$$\zeta = \frac{x - x_{L}}{c} \qquad 4.24$$

$$x_{L} = y\lambda\sigma$$

$$c = c_{m}(1 - \iota\sigma y)$$

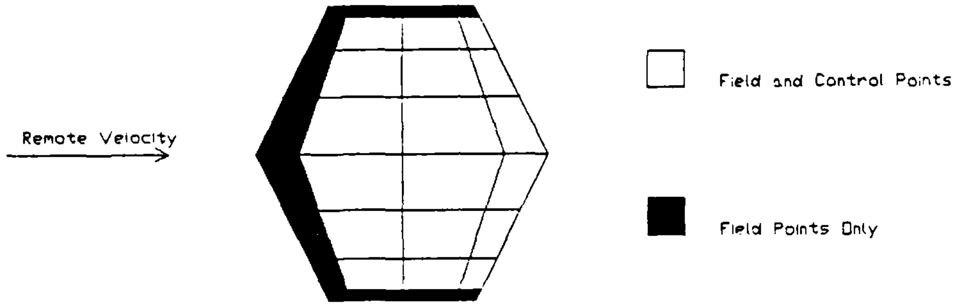
$$\sigma = +1\ on\ right\ wing\ semi-span$$

$$\sigma = +1\ on\ left\ wing\ semi-span$$



4. <u>Field and Control Point Placement</u>

Field and control points are located at the center of the grid elements. When each grid element contains both a field and control point, the circulation that results from solving the equations oscillates wildly and bears no resemblance to the expected solution After considerable probing, a modified arrangement was discovered which generates a reasonable function. Distribution of a "curb" of elements containing only field points along the tips and leading edge gives a satisfactory solution. The remainder of the grid elements contain field and control points. The final configuration is.



Modified Grid Geometry
Figure 4-3

This arrangement gives up some degrees of freedom, but that can be compensated by adding an extra row of grid elements along the chord and an extra column of grid elements along the span.

5. Influence Coefficient Matrices

Equations 4.22 and 4.23 can be divided into multiple integrals. The partial derivatives ($\partial\Gamma/\partial\phi$ & $\partial\Gamma/\partial\theta$) are evaluated at the centers of the elements, and the integrals in 4.22 and 4.23 can be represented as a summation over the elements. Using these assumptions and equations 4.19

through 4.24 it is possible to write the set of equations in matrix form.

The program builds the final matrix of influence coefficients from a number of subsidiary matrices. When equation 4.20 and 4.21 are combined with equation 4.22 and the cosine coordinate transformation we get.

$$\left[M\right]\left\{\Gamma\right\} = \left\{\frac{\partial z}{\partial x_o}\right\} \qquad 4.25$$

Once [M] is generated, it is possible to obtain the circulation solution vector {Γ}.

6. Solving the Set of Equations

The program uses the LEQIF subroutine from the IMSL FORTRAN library as a linear equation solver.

7. Finding the Lift, Drag, and Moment

From the distribution of circulation strength it is possible to find the lift and drag. Barna [Ref.5] and Milne-Thomson [Ref.6] derive the equations for lift and drag, as a function of circulation and down-wash. The program uses these equations to get lift and drag from the circulation on the trailing edge elements. The lift on a span-wise section which lies between y and (y+Δy) is,

$$\Delta L = \rho V \Gamma \Delta y \qquad 4.26$$

and the induced drag on a section is.

$$\Delta D_i = \rho V \Gamma w \Delta y \qquad 4.27$$

Γ is the circulation along the trailing edge and w is the down-wash at the trailing edge. The procedure for finding moment and center of pressure is not as simple. The program would need to resolve forces on the individual elements to find moment coefficient and center of pressure. The CIRCULATION program was abandoned before Fortran code to generate moment coefficient and center of pressure could be written.

C. PROBLEMS ENCOUNTERED

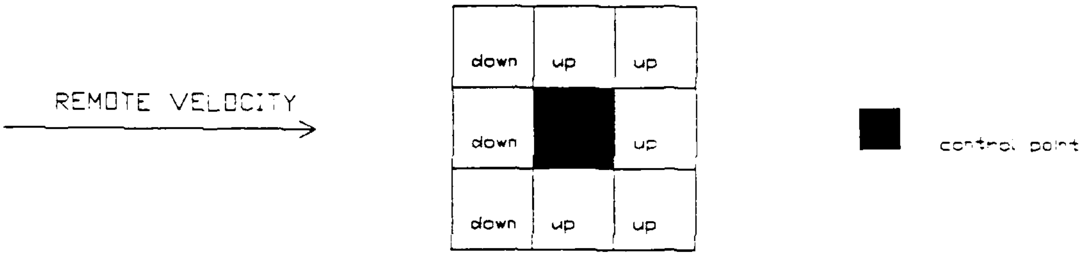
1. Coincident vs Non-coincident Points

The manner used to handle singularities is one of the biggest problems faced when using finite elements to model continuous functions. The circulation model behaves very well when the field and control elements are not coincident.

Field elements induce velocities which increase very rapidly as the distance to the control point decreases. When the field and control points are in the same elemer it appears the induced velocity is indeterminate. However, Milne-Thomson [Ref.6] shows that an element induces zero velocity on itself. This result of the singularity is important and requires special attention.

2. Field and Control Points in the Wing Interior

The direction of velocity induced at a control point by its neighboring field points is significant. Using a simple 3X3 element grid near the center of the wing as an example, Figure 4-4 shows the direction of the induced velocity from 8 field points. All eight field points have positive circulation and surround the center control point.



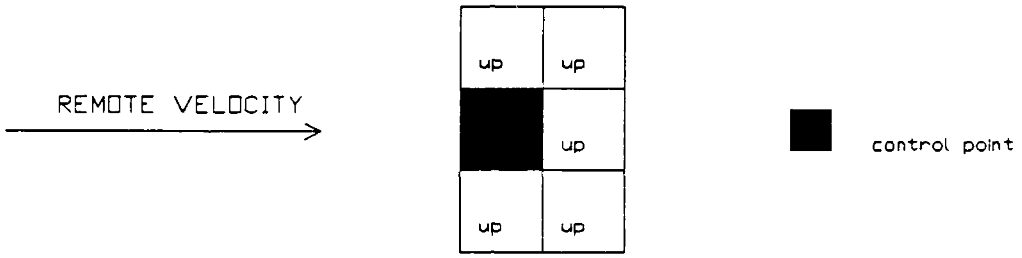
Interior Control Point
Figure 4-4

The three elements upstream from the central control point induce a down-wash on the control point. The other five elements induce an up-wash on the control point. The net velocity is a sum of all eight elements. Recall that the velocity induced is dependant on $\partial\Gamma/\partial x$, not on Γ itself[1]. This net velocity will be down-wash if $\partial\Gamma/\partial x$ in the three upstream elements is sufficiently greater than in the other five elements.

[1] The derivative of circulation is proportional to ΔC_p.

3. Field and Control Points at the Leading Edge

A problem becomes apparent when checking the induced velocity at an element on the leading edge. For a 2X3 grid on the leading edge, Figure 4-5 shows the direction of velocity induced by each of the neighboring field points on the central control point.



Leading Edge Control Point
Figure 4-5

For this situation, all the induced velocity is up-wash, assuming positive circulation. However, the wing requires a net down-wash for flow tangency. Using this arrangement of points, the solution vector oscillates wildly between positive and negative values. This oscillation is a result of the system trying to generate down-wash at the leading edge elements.

The program uses a simple solution to this problem. By ignoring the requirement for flow tangency on the row of elements along the tips and leading edge, the program gives a net down-wash at all other elements. This simplification results in a well behaved, positive circulation over the entire wing.

It appears the set of equations has more unknowns than equations, but boundary conditions provide the additional constraints.

4. Model Complexity

Developing a computer program usable as a teaching tool for basic aerodynamics is the principal goal of the thesis. The CIRCULATION program does not fully achieve that goal. The concept is relatively understandable, but is difficult to execute. The complex method needed to build the influence coefficient matrix reduces its usefulness as a teaching tool. The student needs a simpler tool. The VORTEX program discussed in Section III is that tool.

V. PRESSURE DIFFERENCE MODEL

A. BASIS OF THE ΔC_p MODEL

Like the vortex and circulation models, the pressure difference model (ΔC_p) also uses a governing integral equation as its foundation. The integral equation relates wing slope at a specific point on the wing to the distribution of pressure difference over the whole wing. Dividing the wing into N grid elements, with known slopes, the program solves for N pressure differences (ΔC_p).

The requirements of the Kutta condition are not explicitly satisfied by the equation. Instead, additional constraints are necessary at the wing tips and trailing edge. The simplest form of satisfaction is to require that ΔC_p be zero at those grid elements. That constraint is more severe than necessary. ΔC_p must be zero at the edge of the element but the value at the center of the edge elements can be finite and is approximated by fitting a polynomial through the edge and neighboring control points. This approach is similar to that used in the CIRCULATION model.

B. TECHNIQUE FOR SOLUTION OF THE ΔC_p MODEL

1. The Singularity

The integral equation governing the wing slope/pressure difference relationship is,

$$Z^{\circ}(x,y) = \int\int_S k(x-x_1, y-y_1)\,\Delta C_p(x_1,y_1)\,dx_1\,dy_1 \qquad 5.1$$

where the slope,

$$Z^{\circ}(x,y) = \frac{\partial}{\partial x}\left| Z_m(x,y) \right| \qquad 5.2$$

and,

$$k = \frac{1}{8\pi}\,\frac{1}{(y-y_1)^2}\left| 1 + \frac{(x-x_1)}{\sqrt{(x-x_1)^2+(y-y_1)^2}} \right| \qquad 5.3$$

When the field and control points are coincident *(r=0)* or share the same *y* value *(y=y1)*, there is a strong singularity. It is possible to evaluate this integral and eliminate the singularity.

2. Evaluation of the Integral

Using the mirror image element on the opposite semi-span, it is possible to reduce the integral to,

$$Z^{\circ}(x,y) = \int\int_{S/2} k^{\infty}(x-x_1, y-y_1)\, \Delta C_p(x_1,y_1)\, dx_1\, dy_1 \qquad 5.4$$

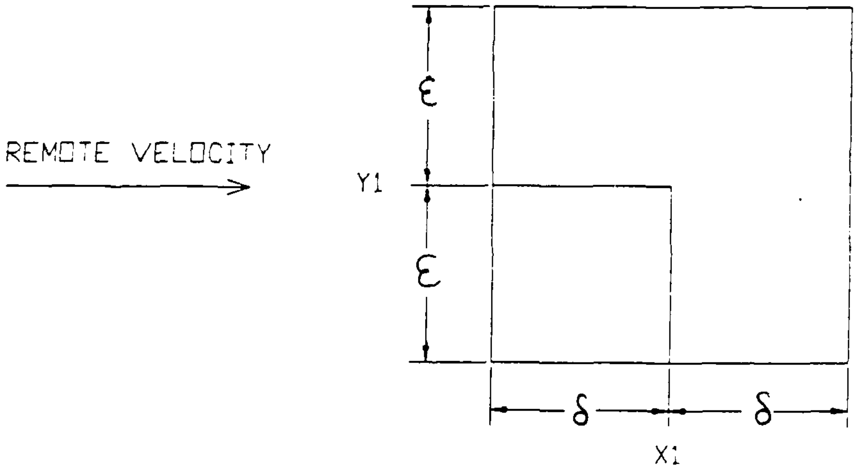
where,

$$k^{\infty} = k_{.} + k^{\circ} \qquad 5.5$$

and

$$k^{\circ} = \frac{1}{8\pi}\, \frac{1}{(y+y_1)^2} \left[1 + \frac{(x-x_1)}{\sqrt{(x-x_1)^2 + (y+y_1)^2}} \right] \qquad 5.6$$

The singularity was eliminated by evaluating the integral. Using Figure 5-1 as a guide, the result is.



Variables used for a Grid Element
Figure 5-1

$$k = \frac{1}{8\pi}(Q_9 + Q_{10} + Q_{11} + Q_{12})$$

$$Q_1 = (x - x_1) + \delta$$

$$Q_2 = (x - x_1) - \delta$$

$$Q_3 = (y - y_1) + \varepsilon$$

$$Q_4 = (y - y_1) - \varepsilon$$

$$Q_5 = \sqrt{Q_1^2 + Q_3^2}$$

$$Q_6 = \sqrt{Q_1^2 + Q_4^2}$$

$$Q_7 = \sqrt{Q_2^2 + Q_3^2}$$

$$Q_8 = \sqrt{Q_2^2 + Q_4^2}$$

$$Q_9 = \frac{4\varepsilon\delta}{Q_3 Q_4}$$

$$Q_{10} = \frac{(Q_6 - Q_8)}{Q_4} - \frac{(Q_5 - Q_7)}{Q_3}$$

$$Q_{11} = \frac{-Q_4}{\left|Q_4\right|} ln\left\{\frac{\left|Q_4\right| + Q_6}{\left|Q_4\right| + Q_8}\right\} + \frac{Q_3}{\left|Q_3\right|} ln\left\{\frac{\left|Q_3\right| + Q_5}{\left|Q_3\right| + Q_7}\right\}$$

$$Q_{12} = \left\{\frac{Q_3 Q_4}{\left|Q_3 Q_4\right|} - 1\right\} ln \left|\frac{Q_1}{Q_2}\right|$$

and k°, the mirror image is;

$$k^\circ = \frac{1}{8\pi}(Q_9 + Q_{10} + Q_{11} + Q_{12})$$

where,

$$Q_3 = (y + y_1) + \varepsilon$$

$$Q_4 = (y + y_l) - \varepsilon$$

and all other values are the same as for k. The end result is,

$$k^{\circ\circ} = k + k^{\circ}$$

and

$$Z^{\circ}(x,y) = \sum_{S/2} k^{\circ\circ} \Delta C_p(x_1,y_1)$$

In matrix form, the equations can be written.

$$\left\{k^{\circ\circ}\right\} \left\{\Delta C_p\right\} = \left\{Z^{\circ}\right\}$$

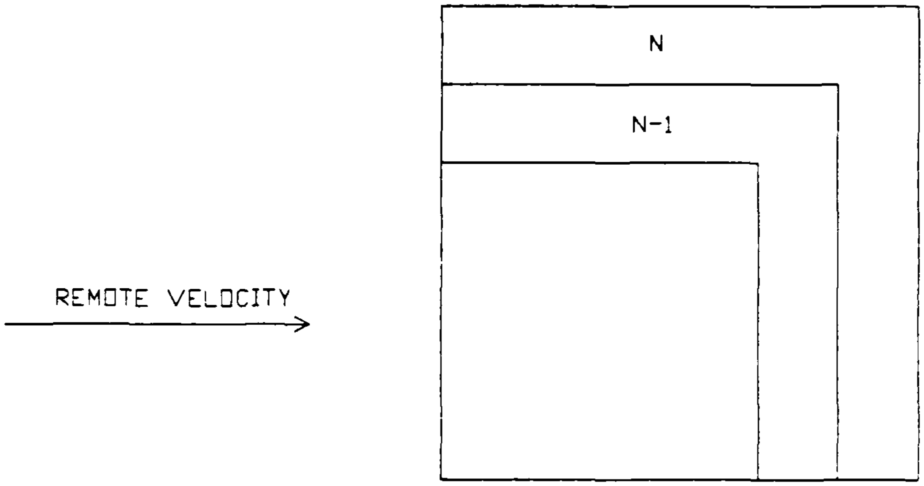
A computer program will rapidly calculate all *of* these values.

3. The Kutta Condition

Solution of the equations does not guarantee a result which satisfies the Kutta condition[1]. To enforce the Kutta condition, the program sets ΔC_p at elements along the

[1] It should be noted that the requirement for flow tangency at the elements near the trailing edge was sufficient to satisfy the Kutta condition in the VORTEX program. *However that result was discovered after the* PRESSURE program was abandoned and no effort was made to extend that reasoning to the PRESSURE program.

wing tip and trailing edge (the nth elements) equal to a fraction of the value of the [n-1]th elements. The program enforces the condition by fitting a polynomial through the two points. See Figure 5-2.



N and N-1th Elements
Figure 5-2

4. Grid Spacing

The program uses a non-uniform grid spacing, based on a cosine function. The elements near the wing tip are small. This spacing has the advantages noted earlier.

C. PROBLEMS ENCOUNTERED

In matrix form, the model has the form.

$$\left|k^{\infty}\right|\left\{\Delta C_{p}\right\}=\left\{Z^{\circ}\right\} \qquad ,\ 5.7$$

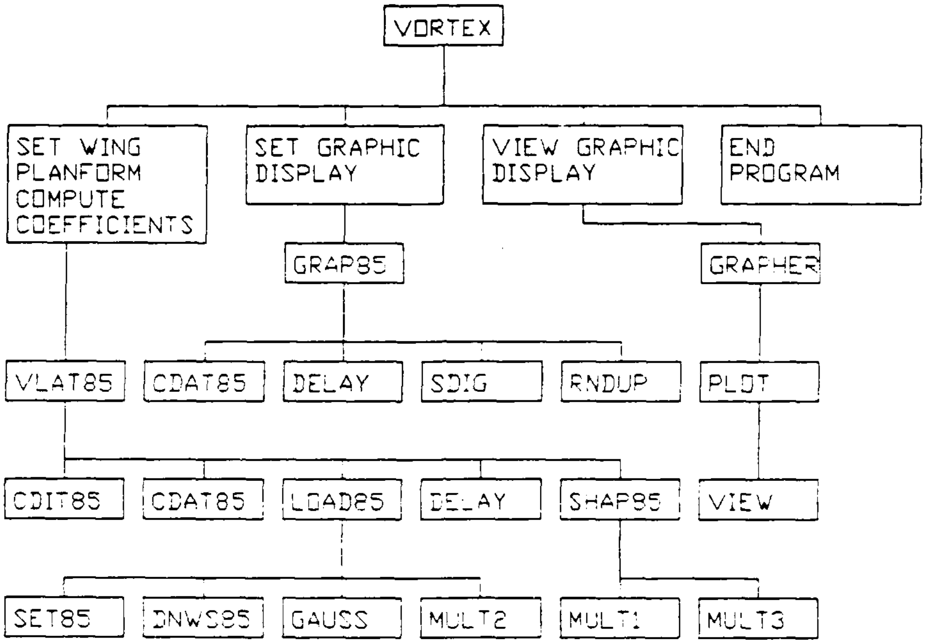
$k^{\circ\circ}$ is the influence coefficient matrix, and z° is the wing slope. The model specifies some values of ΔC_p on the left side of the equality and some values of wing slope ($z°$) on the right side of the equality. This complexity prevents use of the normal matrix solvers. As a result, the solution requires a complex matrix manipulation which is not readily understandable, or desirable for a teaching tool.

D. CONCLUSIONS

The evaluation of the integral over the element has definite advantages for coincident field and control points. The evaluation eliminates a strong singularity and should give a result using finite elements that is very close to the actual property.

The Kutta condition and method used to enforce satisfaction is not optimal. Further studies are needed.

VI. VORTEX PROGRAM FLOW CHART



VII. USERS MANUAL

B. INTRODUCTION

LCDR Chris L. HOLM wrote this program, as a tool for use in AE2035. The program is partial satisfaction of the requirements for the degree Aeronautical Engineer from the Naval Postgraduate School. The objective of the program is to provide a simple computer simulation of flow over a thin wing with low aspect ratio. High aspect ratio wings are better served by a lifting line model. Many advanced programs using elaborate methods to model viscous effects, compressibility, boundary layer growth, and shocks, are available. None of them gives a good introduction to the field of computational aerodynamics. This program should fill that need at the Naval Postgraduate School.

If you wish to skip the theory behind the program, go to section I for instruction on running the program.

There are a number of ways to model the source of forces acting on a body surrounded by fluid in motion. These include potential functions, vortex distributions, circulation distributions, and pressure differential distributions. Each model is related to the other, and each has advantages and disadvantages. This program uses a set of horse shoe vortices to approximate the flow over a wing.

C. SYSTEM REQUIREMENTS

System type	IBM PC or compatible
Memory	256K, minimum
Math Co-processor (8087)	Recommended, not required.
Graphics card	Recommended, not required.
Graphics program	X, Y graphing program like GRAPHER, is recommended, not required.

D. CONCEPT DESCRIPTION

1. The Horse Shoe Vortex

This program, which models the flow by a series of horse shoe vortices, distributed over the wing, is an adaptation of the VORLAT program by Moran [Ref.1]. He develops a program to find the strengths of a series of horse shoe vortices associated with a flat rectangular wing. The VORLAT program has its foundation in two-dimensional airfoil theory, and Moran [Ref.1] adapted the theory to wings of finite aspect ratio.

The Users Manual will present a short description of the VORTEX program. For coverage of the VORLAT program, the foundation of the VORTEX program, consult Moran [Ref.1].

a. Down-wash and the Relationship to Flow Tangency

The velocity induced at a point in the xy plane by a horse shoe vortex, also located in the xy plane, can be determined from:

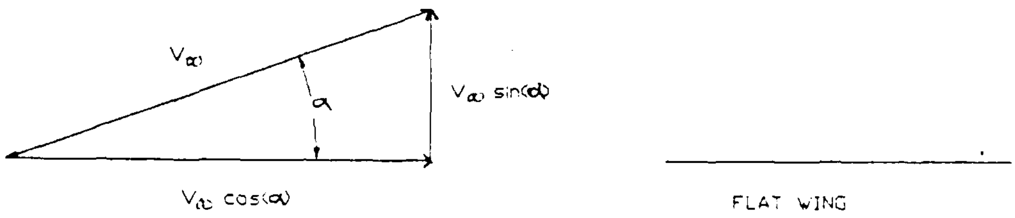
$$w_{ij}(x,y) = \frac{\Delta\Gamma}{4\pi(y-y_a)}\left\{1+\frac{\sqrt{(x-x_a)^2+(y-y_a)^2}}{(x-x_a)}\right\} - \frac{\Delta\Gamma}{4\pi(y-y_b)}\left\{1+\frac{\sqrt{(x-x_a)^2+(y-y_b)^2}}{(x-x_a)}\right\} \qquad 7.1$$

if $(x - x_a)$,

$$w_{ij}(x,y) = \frac{\Delta\Gamma}{4\pi}\left(\frac{1}{y-y_a}-\frac{1}{y-y_b}\right) \qquad 7.2$$

w is the velocity induced at (x,y), called a control point. $\Delta\Gamma$ is the strength of the horse shoe vortex, and (x_a,y_a) and (x_a,y_b) are the corners of the horse shoe vortex. The center of the horse shoe vortex is a field point.

The velocity at a point on a flat wing, which is at an angle of attack α, has components $V_\infty cos(\alpha)$ and $V_\infty sin(\alpha)$ as indicated in Figure 7-1.



Velocity Components on a Flat Wing
Figure 7-1

In order to ensure the flow is tangent to the wing, the induced velocity or down-wash[1] at that point on the wing must also be $V_\infty sin(\alpha)$. Each vortex on the wing contributes to the down-wash[2] at every control point on the wing. So, an equation for each control point, as a function of all the field points can be written. An example of the equation for the control point (x_p, y_p) is.

$$V_\infty sin\alpha - \sum_{ij} \Delta\Gamma_{ij} w_{ij}(x_p, y_p) = 0 \qquad 7.3$$

When the control points are combined, the result is a set of N equations in N unknowns.

b. Placement of the Horse Shoe Vortex

The bound portion of the vortex (that portion perpendicular to the onset flow) is placed at the 1/4 chord point of each grid element on the wing. Flow tangency is evaluated at the 3/4 chord point of each grid element.

[1] Down-wash is considered positive when its direction is downward.

[2] The contribution will be positive when the induced velocity is downward, or negative when the induced velocity is upward.

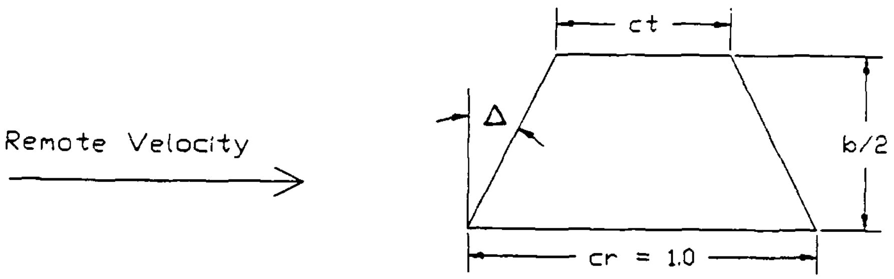
c. Satisfaction of the Kutta Condition

To satisfy the Kutta condition, the vortex strength must go to zero at the trailing edge. For wings and airfoils of zero thickness, flow tangency enforced at the trailing edge appears to be a corollary of the Kutta condition.

2. Developing the Wing

a. Planform Geometry

It is necessary to develop wing geometry from aspect ratio, taper ratio and sweep angle. Aspect ratio is the span divided by the average chord. Taper ratio is the tip chord divided by the root chord, and sweep angle is the angle between the leading edge and the y axis. Moran [Ref.1] uses a fixed root chord and varies the span as necessary to achieve the required aspect ratio and taper ratio. The VORTEX program uses this approach, which works well.



Planform and Variables used in VORTEX Program
Figure 7-2

Aspect ratio, AR, is (b^2/S). Span (b) is the distance from wing tip to wing tip, and area (S) is the total planform area of the wing.

Taper ratio, λ, is the ratio of tip chord to root chord.

$$\lambda = \frac{tip\ chord}{root\ chord} = \frac{c_t}{c_r} \qquad 7.4$$

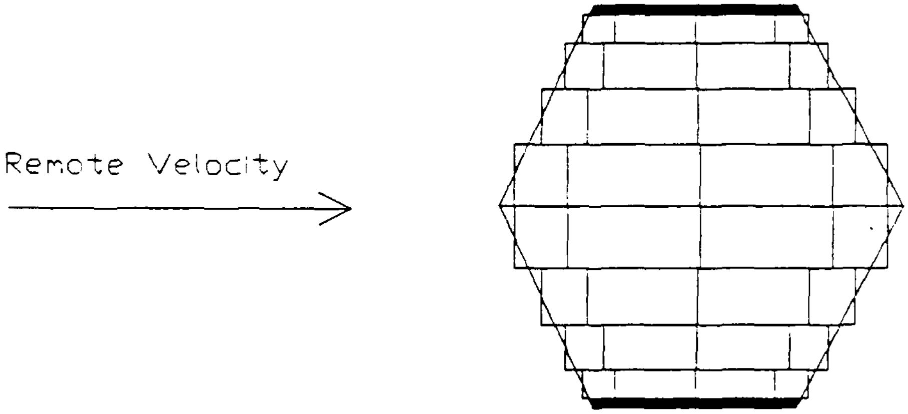
Sweep angle delta, Δ, is the angle that the leading edge makes with a line drawn perpendicular to the root chord.

b. **Grid Geometry**

Using aspect ratio, taper ratio, and sweep angle, the subroutine SET85 develops a grid. The program stores the coordinates of all necessary points in the matrices WING and SECTN. Results of the program are also stored in WING and SECTN which are written to data files when the program finishes.

In a manner analogous to that used by Hough [Ref.3], the tip vortices are inset to improve accuracy of the results. The inset distance is one element wide.

The model keeps matrix size and the number of grid elements small to keep the time necessary for solution within reason. The model concentrates grid elements in areas where the rate of change is rapid, or where the values are most important. Those areas are along the wing boundaries. An excellent method for concentrating the points, while keeping the total number of points at a minimum, is through cosine spacing. Figure 7-3 contains a sample of the layout. Notice that the rectangular elements do not model the planform exactly, but in the limit the difference will be negligible.



Grid Model with Cosine Spacing
Figure 7-3

3. Matrices Used in VORTEX Program

The program uses two identical matrices of influence coefficients, [A] and [AA]. [A] is used to solve for the $\Delta\Gamma$ of the flat wing, and [AA] is used to solve for the wing shape of the elliptically loaded wing.[3]

$$\left[A \right]^{-1} \left\{ \frac{dz}{dx} \right\} = \left\{ \Delta\Gamma \right\} \quad \textit{(Flat Wing)} \qquad 7.5$$

$$\left[AA \right] \left\{ \Delta\Gamma \right\} = \left\{ \frac{dz}{dx} \right\} \quad \textit{(Elliptically Loaded Wing)} \qquad 7.6$$

The SECTN matrix contains values for the wing sections, (chord, and width) as well as some of the final results. Each row of the SECTN matrix corresponds to a wing section in the semi-span. The Output Variables section (H.1.a) describes the columns.

The WING matrix contains wing coordinates required by the program as well as some of the final results. Each row of the matrix corresponds to an element in the wing semi-span. The Output Variables section (H.1.b) describes the columns.

[3] The method used to develop wing shape will be discussed in section VII.E.

4. Solution of the Set of Equations

The subroutine GAUSS solves the N equations in N unknowns. The subroutine destroys [A] in the process but returns the solution vector ($\Delta\Gamma$) in pl-ce of the right hand sides.

5. Finding the Forces on the Elements

The Kutta-Joukowski theorem states that the force per unit span acting on an element is.

$$\frac{\overrightarrow{\Delta F}}{\Delta y} = \rho \overrightarrow{V_{eff}} \ x \ \overrightarrow{\Delta\Gamma} \qquad 7.7$$

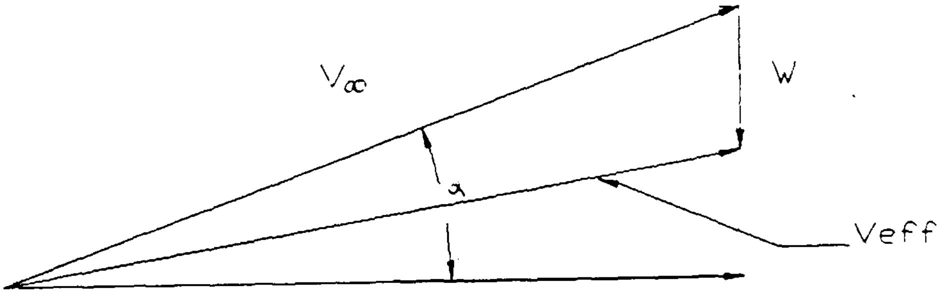
In this equation, $\overline{V_{eff}}$ is the local effective velocity at the center of the element. $\overline{V_{eff}}$ is defined in equation 7.8 and Figure 7-4. ρ is the density and $\Delta\Gamma$ is the incremental circulation around the element This circulation ($\Delta\Gamma$) is the same $\Delta\Gamma$ contained in equation 7.1. $\Delta\Gamma$ can be termed the incremental circulation that occurs over the element.

Throughout this section, certain approximations and dimensional simplifications will be made. The first is the small angle approximation, where the sine is approximately equal to the angle in radians, and the cosine is approximately equal to 1.0. This approximation requires that angle of attack for the flat wing be small, generally less than 10 degrees. The approximation also requires that any wing slope on the elliptically loaded wing be small. This requirement is met by restricting the desired lift coefficient to values less than about 0.5. In the process of determining coefficients of lift, drag and moment, certain dimensional reference quantities arise. These quantities are density (ρ), remote velocity (V_∞), planform area (S), and average chord ($\bar{c}$). In performing dimensional analysis an arbitrarily unit value can be assigned to these terms.

a. **Finding ΔC_p**

The distribution of ΔC_p over the wing is desired, so the force on each element must be converted to a dimensionless pressure difference coefficient. $\overline{V_{eff}}$ is defined as.

$$\overrightarrow{V_{eff}} = V_\infty \cos\alpha \, \vec{i} + \left(V_\infty \sin\alpha - w \right) \vec{j} \qquad 7.8$$



Components of Effective Velocity ($\overline{V_{eff}}$)
Figure 7-4

It is also significant to note that ΔΓ can be written as,

$$\overrightarrow{\Delta\Gamma} = \Delta\Gamma \vec{j} \qquad 7.9$$

since the bound portion of the horse shoe vortex is aligned with the y axis. The result of the force cross product, Equation 7.7, is.

$$\frac{\overrightarrow{\Delta F}}{\Delta y} = \rho \left| \left(w - V_\infty sin\,\alpha \right)\Delta\Gamma \vec{i} + V_\infty cos\,\alpha \Delta\,\Gamma \vec{k} \right| \qquad 7.10$$

Setting $\rho = 1$, $V_\infty = 1$, and incorporating the small angle approximation.

$$\frac{\overrightarrow{\Delta F}}{\Delta y} = \left| \left(w - \alpha \right)\Delta\Gamma \vec{i} + \Delta\,\Gamma \vec{k} \right| \qquad 7.11$$

This force is related to ΔC_p. ΔC_p is a scalar, rather than a vector quantity, so the program uses the force component normal to the wing to compute ΔC_p. That component is.

$$\Delta F = \Delta y \Delta\Gamma \qquad 7.12$$

Making the force non-dimensional gives a pressure difference coefficient, ΔC_p.

$$\Delta C_p = \frac{\Delta F}{\frac{1}{2}\rho V_\infty^2 \Delta x \Delta y} \qquad 7.13$$

Finally, after setting $\rho = 1$ and $V_\infty = 1$, ΔC_p for an element is.

$$\Delta C_p = \frac{2\,\Delta F}{\Delta x \Delta y} = \frac{2\,\Delta\Gamma}{\Delta x} \qquad 7.14$$

This provides the ΔC_p for each element, which can be plotted versus chord, for each span-wise section.

b. Finding the Lift, Drag and Moment

Lift, drag and moment are also found from the vector force on each element. Equation 7.10 showed that the force on an element has components normal and tangent to the wing. Once more setting $\rho = 1$, and $V_\infty = 1$, the vector force on an element is.

$$\overrightarrow{\Delta F} = \left|\left(w - \sin a\right)\Delta\Gamma\Delta y\,\vec{i} + \cos a\,\Delta\Gamma\Delta y\,\vec{k}\right| \qquad 7.15$$

These forces are oriented relative to the wing coordinate system, where $\vec{i}$ is parallel to the wing and $\vec{k}$ is perpendicular to the wing. Lift and drag are normal and tangent, respectively, to the remote velocity (V_∞). The wing is at an angle of attack (α) to that remote velocity. Using this information, the program transforms the forces to a coordinate system oriented to the remote velocity. For an individual element the lift and drag forces are;

$$\Delta L = Elemental\,Lift\,Force = (\cos\alpha)(\Delta y\Delta\Gamma)(\cos\alpha) - (w - \sin\alpha)(\Delta y\Delta\Gamma)(\sin\alpha)$$

$$\Delta D = Elemental\,Drag\,Force = (\cos\alpha)(\Delta y\Delta\Gamma)(\sin\alpha) + (w - \sin\alpha)(\Delta y\Delta\Gamma)(\cos\alpha)$$

using a small angle approximation, 7.16

$$\Delta L = Elemental\,Lift\,Force = \Delta y\Delta\Gamma - w\alpha\Delta y\Delta\Gamma = \Delta y\Delta\Gamma$$

$$\Delta D = Elemental\,Drag\,Force = \Delta y\Delta\Gamma\alpha + (w - \alpha)\Delta y\Delta\Gamma = w\Delta y\Delta\Gamma$$

The total lift and total drag acting on the wing is a summation of the elemental lifts and drags. Using wing symmetry;

$$L = Total\,Lift = 2\sum_{semi-span}\Delta L$$

7.17

$$D = Total\,Drag = 2\sum_{semi-span}\Delta D$$

The moment about the y axis for an individual element, with the accepted sign convention, is;

$$\Delta M = Elemental\,Moment = -(normal\,wing\,force)\,(x\,position\,of\,center\,of\,element)$$ 7.18

$$\Delta M = Elemental\,Moment = (-\Delta y\Delta\Gamma)(x_{cen})$$

Total moment about the y axis is a summation of the elemental moments. Once again, using wing symmetry.

$$M = Total\,Moment = 2 \sum_{semi-span} \Delta M \qquad 7.19$$

6. Finding the Coefficients

Non-dimensionalizing the total lift and drag gives,

$$C_L = \frac{L}{\frac{1}{2}\rho V_\infty^2 S}$$

$$C_D = \frac{D}{\frac{1}{2}\rho V_\infty^2 S} \qquad 7.20$$

where S = span x average chord = (b x $\bar{c}$), ρ = 1 and V_∞ = 1.

Non-dimensionalizing the total moment about the y axis gives,

$$C_{Mo} = \frac{M}{\frac{1}{2}\rho V_\infty^2 \bar{c} S} \qquad 7.21$$

where $\bar{c}$ is the average chord.

The aerodynamic center X_{ac} is important. It is the chord-wise position about which the flat wing has zero moment. The program finds X_{ac} from.

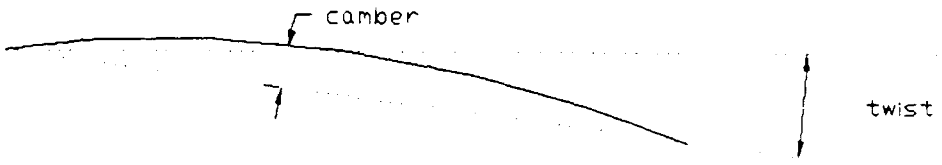
$$X_{ac} = \frac{-\,total\,moment}{total\,lift} \qquad 7.22$$

This assumes that total moment is taken about the y axis, or x=0. The program finds the moment on the elliptically loaded wing, and uses the X_{ac} from the flat wing as the reference axis to find C_{Mac}.

E. TECHNIQUE FOR SPECIFIED LOADING

1. Finding Camber and Twist from Loading

Elliptic loading in a span-wise direction results in minimum induced drag, and loading is proportional to circulation. At sufficiently high aspect ratio, a flat wing with elliptic area distribution generates elliptic loading but weighs more than a straight wing. Wings with straight leading and trailing edges also cost less to manufacture than elliptic wings. So, a span-wise elliptic lift distribution is generated by slightly twisting the wing. The direct relationship between wing shape (camber and twist) and circulation distribution makes this generation possible.



Camber and Twist on the Elliptically Loaded Wing
Figure 7-5

For a thin wing, elliptic load distribution in the chord-wise direction requires an approximately parabolic camber shape. The program specifies an elliptic chord-wise load distribution and then finds the associated shape.

2. Specifying the Lift Coefficient

Specifying the form and amplitude of the ideal lift distribution over the wing determines the form and amplitude of the wing shape. A load distribution of elliptic **form** is imposed in this case and the corresponding **amplitude** is fixed by the desired ideal wing lift coefficient, C_{Li}.

3. Maximum Desired Lift Coefficient

The small angle approximation breaks down when large lift coefficients are specified. To prevent this occurrence, the maximum desired lift coefficient is restricted to 0.5.

4. Achieving the Desired Lift Coefficient

The program scales a reference distribution of ΔC_p, which is elliptic in form, to achieve the desired total lift coefficient for the wing.

5. Forces, Moments and Coefficients

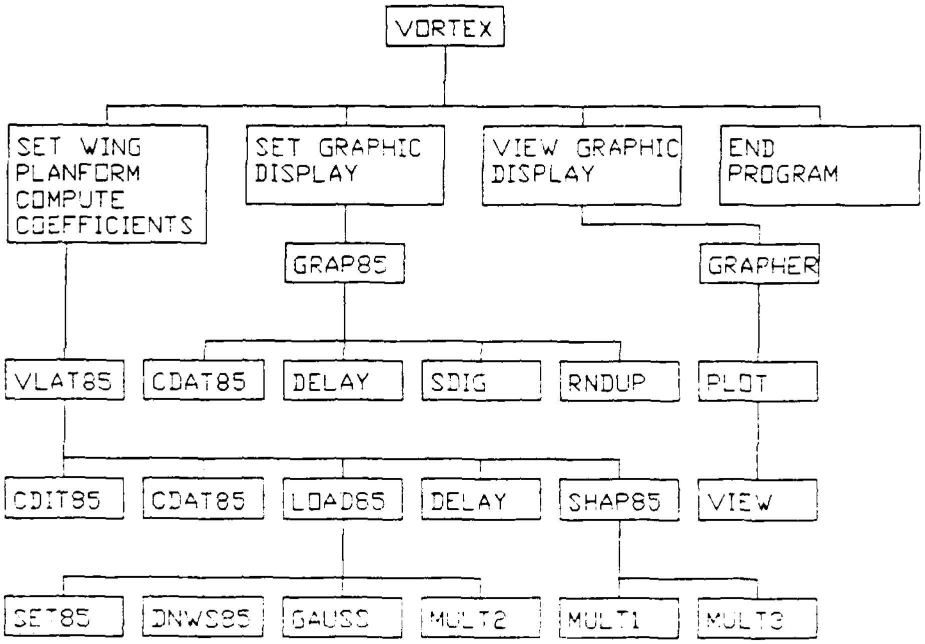
Once the ΔC_p distribution is scaled for the elliptically loaded wing, the program finds forces, moments and coefficients in the same manner as for the flat wing. The only difference is in the coordinate transformation used to convert forces on the wing to lift and drag. The individual elements on the elliptically loaded wing are no longer at a uniform angle to the remote velocity. This change in angle must be taken into account when performing the coordinate transformation which gives lift and drag forces on individual elements.

The program finds moment coefficient about the aerodynamic center, $C_{M_{ac}}$, from,

$$C_{M_{ac}} = C_{M_o} + C_{L_i}\left(\frac{X_{ac}}{\bar{c}}\right) \qquad 7.23$$

where X_{ac} is found from the flat wing, $\bar{c}$ is the average chord. C_{M_o} is the pitching moment of the elliptically loaded wing about the y axis, and C_{L_i} is the lift coefficient of the elliptically loaded wing.

F. FLOW CHART FOR MODULES



G. INPUT VARIABLES

There are eight input variables. They are:

1. Aspect Ratio. AR, defined as b^2/S, where b is the total span and S is the total area, can have any positive value less than 20.0. Higher aspect ratios should use lifting line theory.
2. Taper Ratio. LAMDA, defined as C_t/C_r, where C_t is the wing tip chord and C_r is the wing root chord, can have any positive value between 0 and 1.
3. Leading Edge Sweep. DELTA, defined as the angle in degrees that the leading edge of the wing makes with a line perpendicular to the remote velocity, can have any positive value between 0 and 60.
4. Angle of Attack. ALPHA, defined as the angle in degrees made between the flat wing and the remote velocity, can have any positive value between 0 and 10.
5. Number of elements in a section. NX, defined as the integer number of elements in a chord of the wing, can have any value between 1 and 10.
6. Number of elements in a semi-span. NY, defined as the integer number of elements in a semi-span of the wing, can have any value between 1 and 10.
7. Desired lift coefficient. CLDSRD, the desired lift coefficient for the elliptically loaded wing, can have any value between 0 and 0.5.

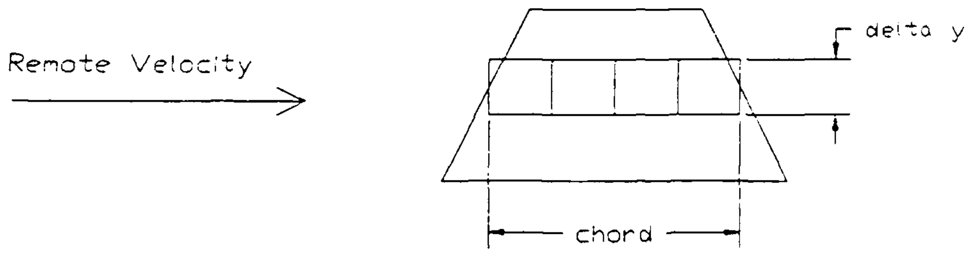
H. OUTPUT VARIABLES

1. Tabular Data

When complete, the program writes four tabular data files. One file, the SECTION.MAT file, contains data for each section in the wing semi-span. Another, the WING.MAT file, contains data for the individual grid elements on the wing semi-span. The third, the FLAT.DAT file, shows the coefficients for the flat wing. The fourth, the CAMBER.DAT file, shows the coefficients for the cambered or elliptically loaded wing.

a. SECTION.MAT

Each row, or line, in the file represents a section of the wing semi-span. Each column represents a different variable within the section.



A Single section in the Wing Semi-Span
Figure 7-6

The columns are:

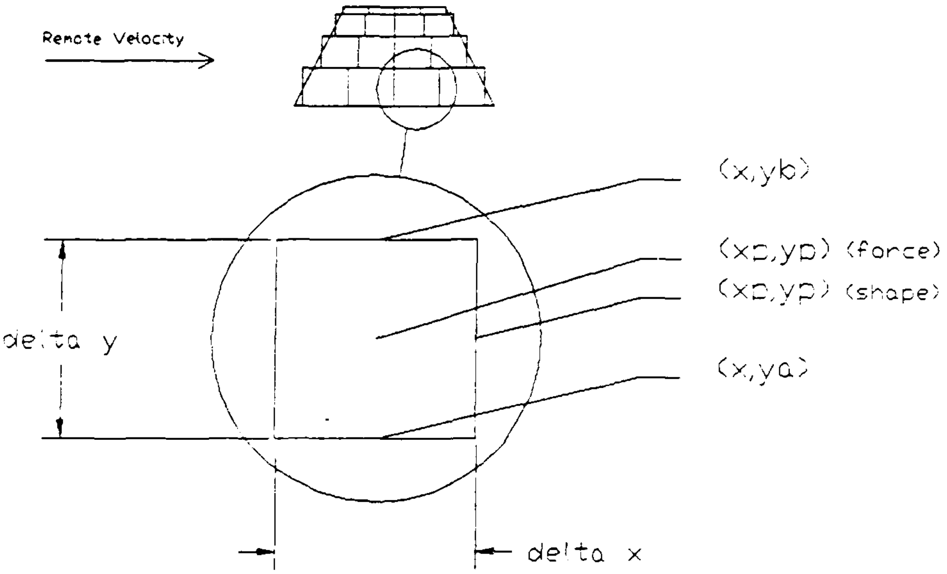
1. Section chord length
2. Section lift coefficient per unit span for the flat wing. The integral of these values from tip to tip is the total lift coefficient.
3. Section drag coefficient per unit span for the flat wing. The integral of these values from tip to tip is the total drag coefficient.
4. Section moment coefficient per unit span about the aerodynamic center for the flat wing. The integral of these values from tip to tip is the total moment coefficient about the aerodynamic center.
5. Section x_{ac}, aerodynamic center relative to x=0, for the flat wing.
6. Section delta y
7. Section lift coefficient per unit span for the cambered wing. The integral of these values from tip to tip is the total lift coefficient.
8. Section drag coefficient per unit span for the cambered wing. The integral of these values from tip to tip is the total drag coefficient.
9. Section moment coefficient per unit span about the aerodynamic center for the cambered, or elliptically loaded wing. The integral of these values from tip to tip is the total moment coefficient about the aerodynamic center.

10. Section twist in degrees of the cambered wing.
Positive angle of twist is nose up.

The program arranges the rows from root to tip, with the root section on the first line and the tip section on the last line.

b. **WING.MAT**

Each row, or line in the file, represents an individual element in the wing grid. Each column represents a different variable for that element. Figure 7-7 shows the location of some variables on a representative grid element.



A Single Element in the Wing Semi-Span
Figure 7-7

The columns are:

1. Sequential integer identifier of the element, also called IJ in the program.
2. Integer y position of the element. 1 is at the wing root, NY is at the tip.
3. Integer x position of the element. 1 is at the leading edge, NX is at the trailing edge.
4. x position of center of element.
5. y position of center of element.
6. x position of horse shoe vortex, a field point.
7. Blank, or zero.
8. ya position of one corner of horse shoe vortex.
9. yb position of one corner of horse shoe vortex.
10. xp trailing edge of the element, a control point used in computing wing shape or $\Delta\Gamma$.
11. xp center of the element, a control point used in computing the forces acting on the wing.
12. yp center of the element, a control point.
13. Delta x, chord-wise dimension of individual element.
14. Delta y, span-wise dimension of individual element.
15. Flat wing incremental circulation ($\Delta\Gamma$).
16. Flat wing ΔC_p.
17. Cambered wing incremental circulation ($\Delta\Gamma$), adjusted for $C_L$ref.
18. Cambered wing ΔC_p
19. Cambered wing slope ($\partial z/\partial x$).

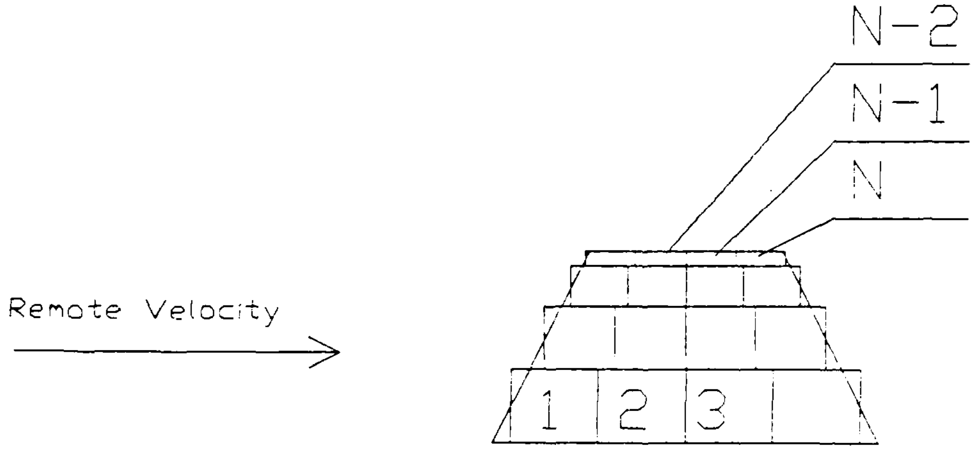
20. Cambered wing height (z).

21. Cambered wing incremental circulation ($\Delta\Gamma$), if $C_L$ref were = 1.0.

22. Down-wash (w) at center of flat wing element due to trailing filaments.

23. Down-wash (w) at center of cambered wing element due to trailing filaments.

Some columns are identical. Separate columns were used for each variable during the development phase to promote ease in modification. The rows, or lines, of the WING.MAT file are arranged as follows.



Grid Element Numbering on the Semi-Span
Figure 7-8

The first row is the leading edge element in the root section. The next element is immediately behind the leading edge and so on. The last row is the trailing edge element in the wing-tip section.

c. FLAT.DAT

This file contains coefficients and results for the flat wing. An example is shown.

```
FLAT WING
CL      =      .177721
CD      =      .006516
CD/CL2  =      .206312
CMAC    =      .000000
XAC     =      .279935
AR      =     1.330000
LAMDA   =      .500000
DELTA   =    25.000000
NX      =   8
NY      =   8
ALPHA   =     5.000000
CLDSRD  =      .200000
ELLIP   = YES
CLa, Lift Curve Slope
per/Deg=       .035544
```

d. CAMBER.DAT

This file contains coefficients and results for the cambered, or elliptically loaded, wing. An example is shown.

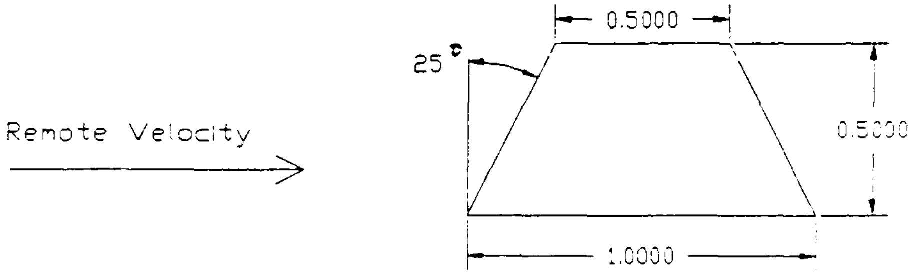
```
CAMBERED WING
CL      =      .199311
CD      =      .009040
CD/CL2  =      .227557
CMAC    =     -.062732
AR      =     1.330000
LAMDA   =      .500000
DELTA   =    25.000000
NX      =   8
NY      =   8
ALPHA   =     5.000000
CLDSRD  =      .200000
ELLIP   = YES
```

I. SAMPLE PROBLEM

A sample problem will illustrate use of the VORTEX program. A wing planform with the following characteristics will be analyzed.

Aspect ratio	1.333
Taper ratio	0.5
Leading Edge Sweep	25 degrees
Angle of Attack	5 degrees
Number of span elements	8
Number of chord elements	8
Desired Lift Coefficient	0.2

The planform looks like Figure 7-9.



Planform used in the Sample Problem
Figure 7-9

1. Starting the Program

After turning the computer on go to the DOS prompt, which generally looks like this.

```
C:>
```

Change the program to the VORTEX directory by typing

CD\VORTEX [return]

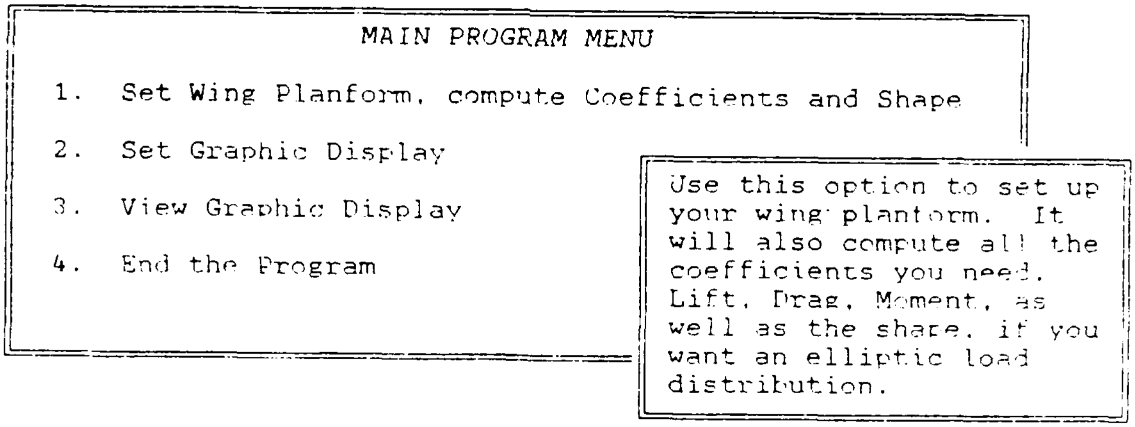
the screen should now look like.

```
C\VORTEX:>
```

To start the program, type

VORTEX [return]

The program will start and the following menu will be displayed.



```
                    MAIN PROGRAM MENU
  1.  Set Wing Planform, compute Coefficients and Shape
  2.  Set Graphic Display
  3.  View Graphic Display
  4.  End the Program
```

```
Use this option to set up
your wing planform.  It
will also compute all the
coefficients you need.
Lift, Drag, Moment, as
well as the shape, if you
want an elliptic load
distribution.
```

Main Program Menu
Figure 7-10

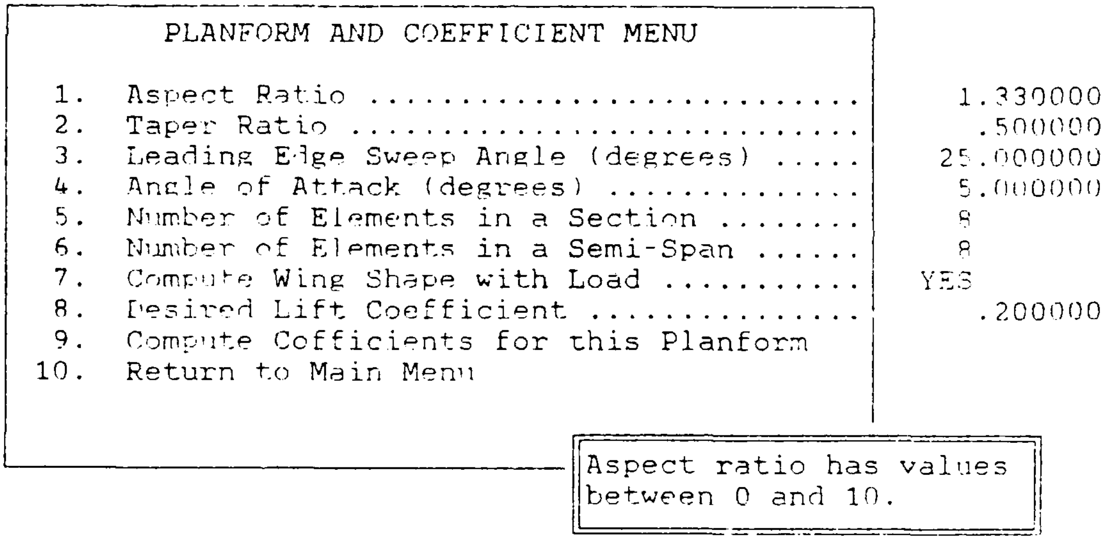
2. Main Program Menu

There are four options in the MAIN PROGRAM MENU. All coefficients and results are computed in option 1. Whenever the planform is changed, option 1 must be selected to recompute the coefficients. The coefficients and results are saved to data files after being computed in option 1. The data files allow you to use options 2 and 3 as often as desired without re-computing the results for the planform.

Help menus are available. Press F1 to toggle the help menus on or off.

3. Planform and Coefficient Menu

The program saves planform variables and results from the most recent solution and shows them in the PLANFORM and COEFFICIENT MENU. The menu looks like Figure 7-11.



```
                 PLANFORM AND COEFFICIENT MENU

   1.  Aspect Ratio ............................      1.330000
   2.  Taper Ratio .............................       .500000
   3.  Leading Edge Sweep Angle (degrees) .....     25.000000
   4.  Angle of Attack (degrees) ..............      5.000000
   5.  Number of Elements in a Section ........      8
   6.  Number of Elements in a Semi-Span ......      8
   7.  Compute Wing Shape with Load ...........     YES
   8.  Desired Lift Coefficient ...............       .200000
   9.  Compute Cofficients for this Planform
  10.  Return to Main Menu

                                          Aspect ratio has values
                                          between 0 and 10.
```

Planform and Coefficient Menu
Figure 7-11

Options 1, 2, and 3 change Aspect Ratio, Taper Ratio, and Leading edge sweep angle, respectively.

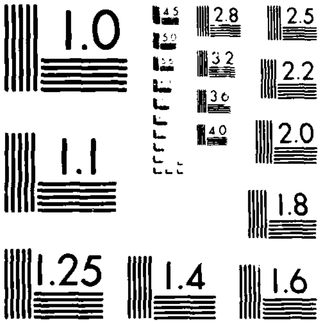
Option 4 changes the angle of attack.

Option 5 changes the number of chord-wise grid elements, and

AD-A200 436 AERODYNAMIC PERFORMANCE OF WINGS OF ARBITRARY PLANFORM 2/3
IN INVISCID INCOMPRESSIBLE IRROTATIONAL FLOW(U) NAVAL
POSTGRADUATE SCHOOL MONTEREY CA C L HOLM SEP 88

UNCLASSIFIED F/G 1/1 NL

UNCLAS



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963 A

Option 6 changes the number of span-wise grid elements in a semi-span.

Option 7, Compute Wing Shape with Load, is YES or NO. Because the time required to compute wing shape is extensive, that option may not be desirable for all problems.

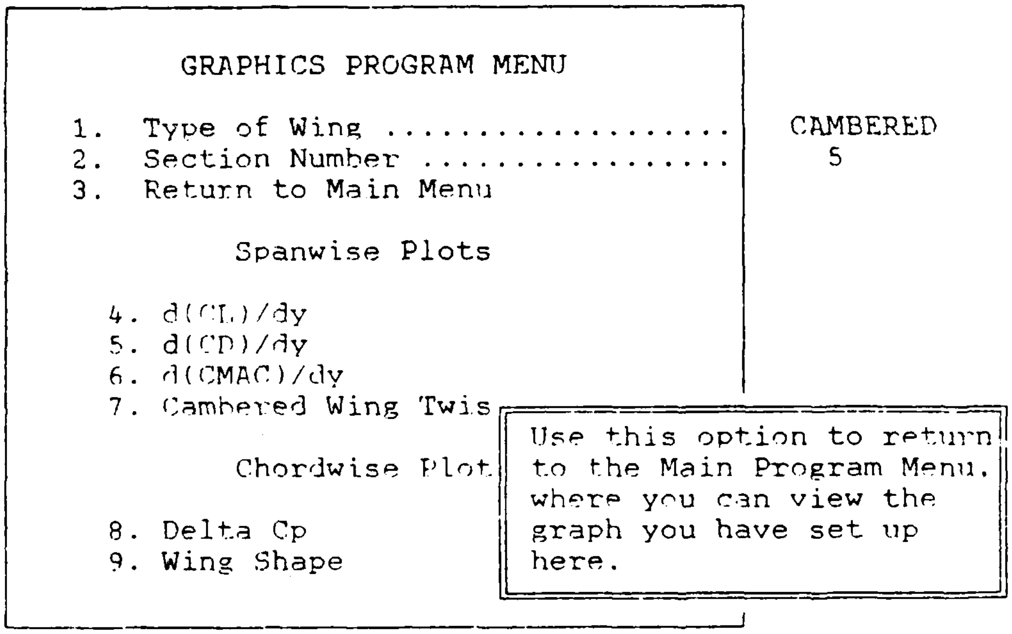
Option 8 changes the desired lift coefficient, C_{Li} of the elliptically loaded wing.

Option 9 computes the coefficients. The coefficients are written to four files, SECTION.MAT, WING.MAT, FLAT.DAT, and CAMBER.DAT. Section H of this thesis contains a description of the elements in the files.

Option 10 returns to the Main Program Menu to view the results.

4. Graphics Program Menu

Choosing selection 2 of the Main Program Menu displays the Graphics Program Menu. An example is shown in Figure 7-12.



```
        GRAPHICS PROGRAM MENU

 1.  Type of Wing ...................    CAMBERED
 2.  Section Number .................      5
 3.  Return to Main Menu

            Spanwise Plots

   4. d(CL)/dy
   5. d(CD)/dy
   6. d(CMAC)/dy
   7. Cambered Wing Twis

            Chordwise Plot

   8. Delta Cp
   9. Wing Shape
```

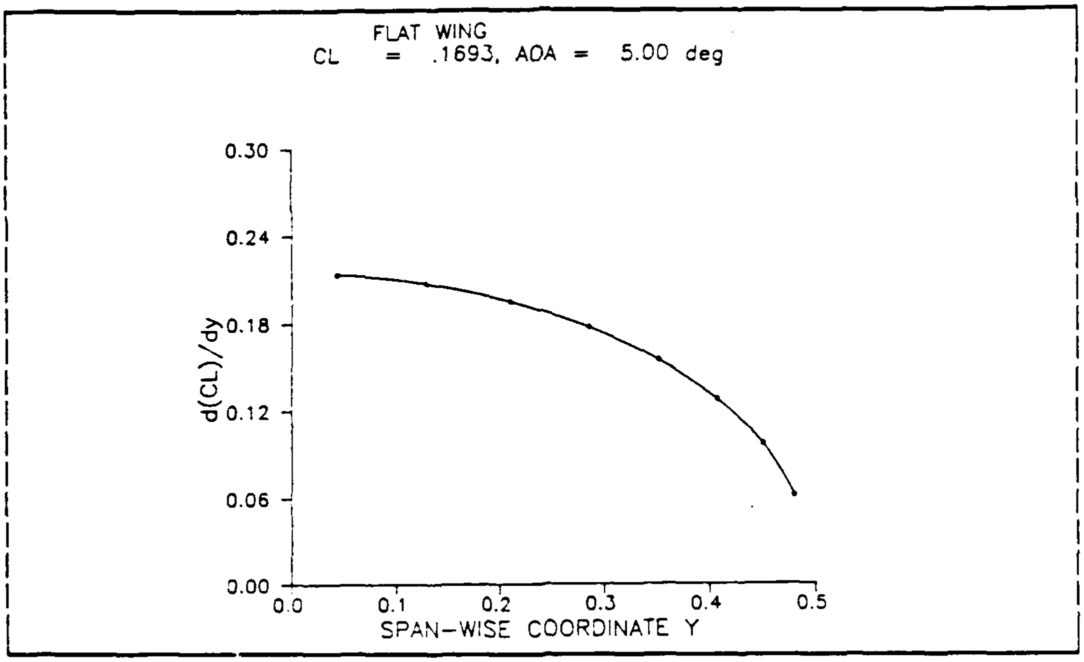
Use this option to return to the Main Program Menu, where you can view the graph you have set up here.

Graphics Program Menu
Figure 7-12

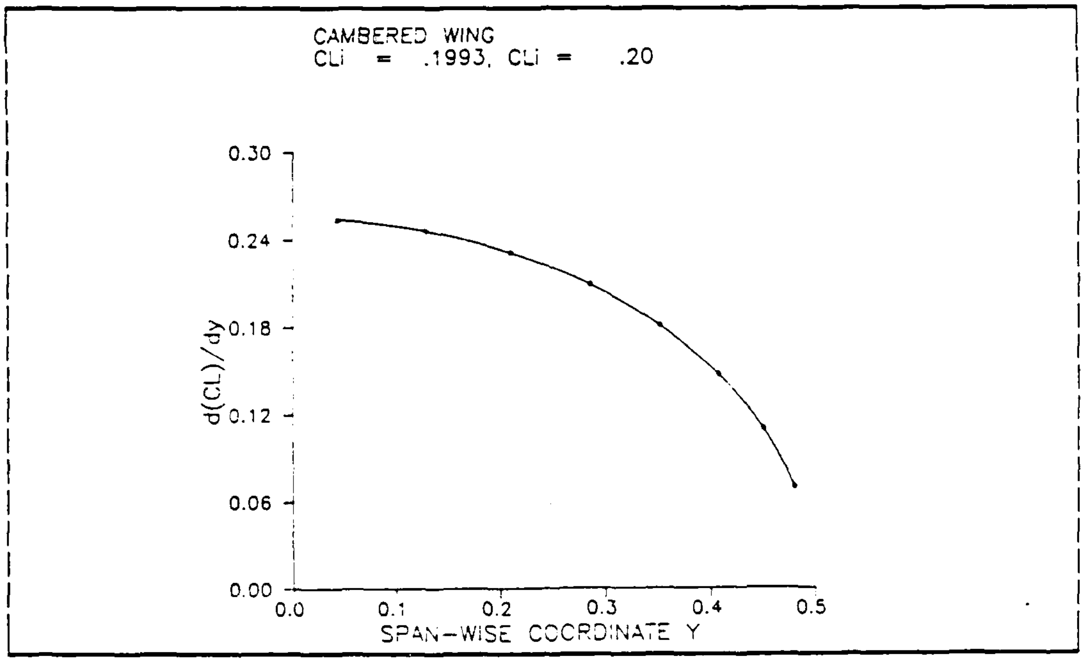
Option 1 changes the type of wing that will be displayed, either flat or cambered.

Option 2 changes the section of the wing which is plotted with options 8 and 9.

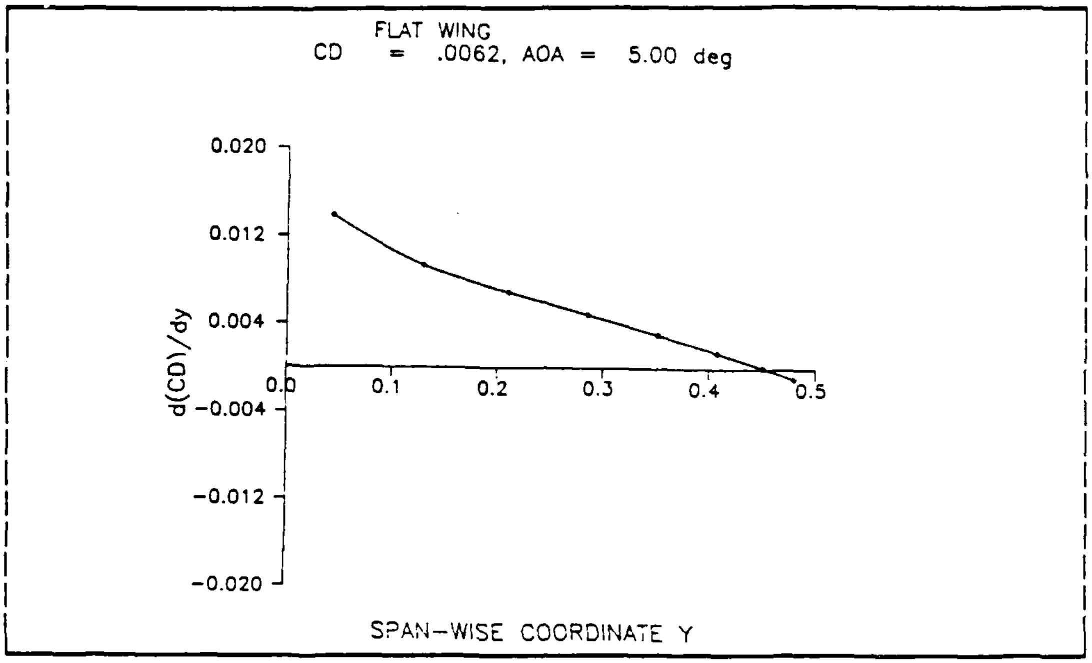
After setting options 1 and 2, select the desired plot from options 4 through 9. The necessary data files will be prepared and the program will return to the Main Program Menu, where the selected plot can be seen using the View Graphic Di:play option. Examples follow.



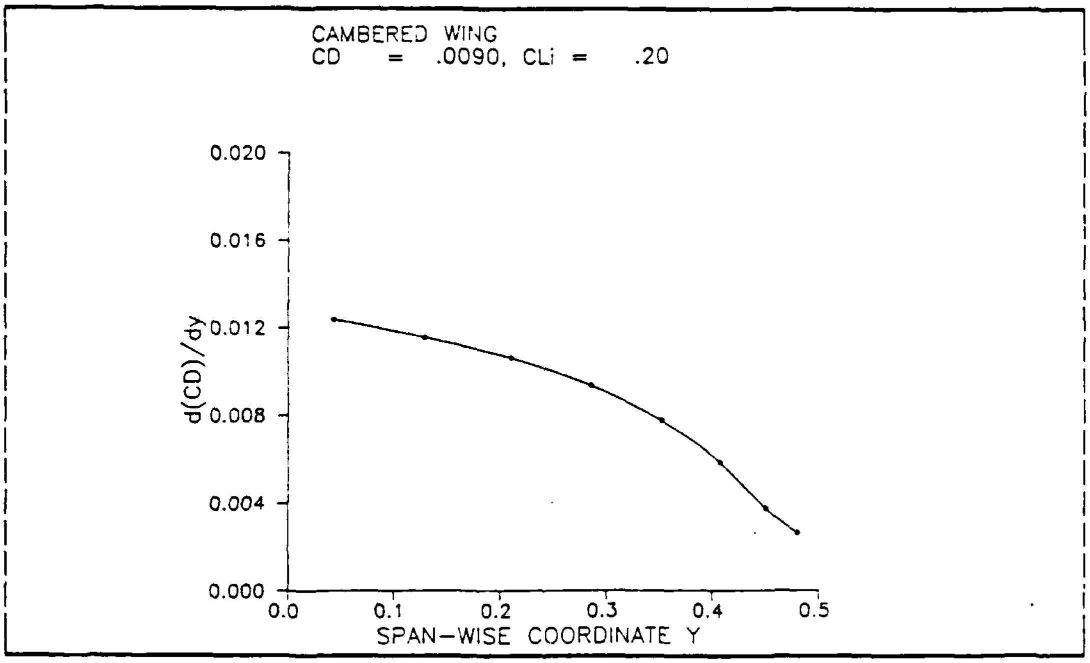
FLAT WING d(CL)/dy vs SPAN
Figure 7-13



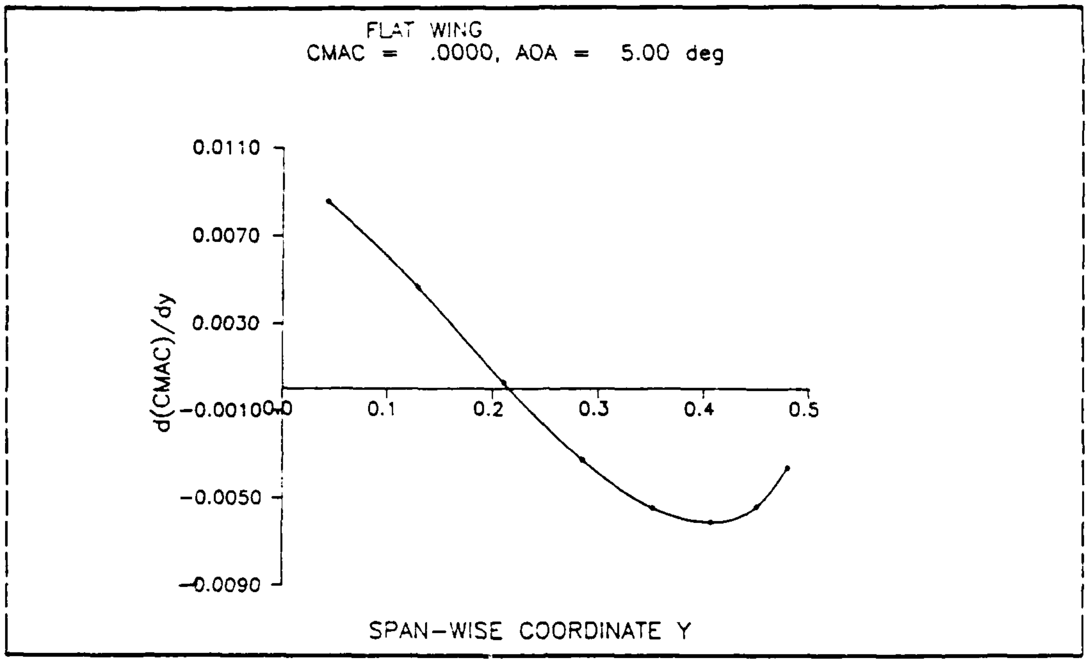
CAMBERED WING d(CL)/dy vs SPAN
Figure 7-14



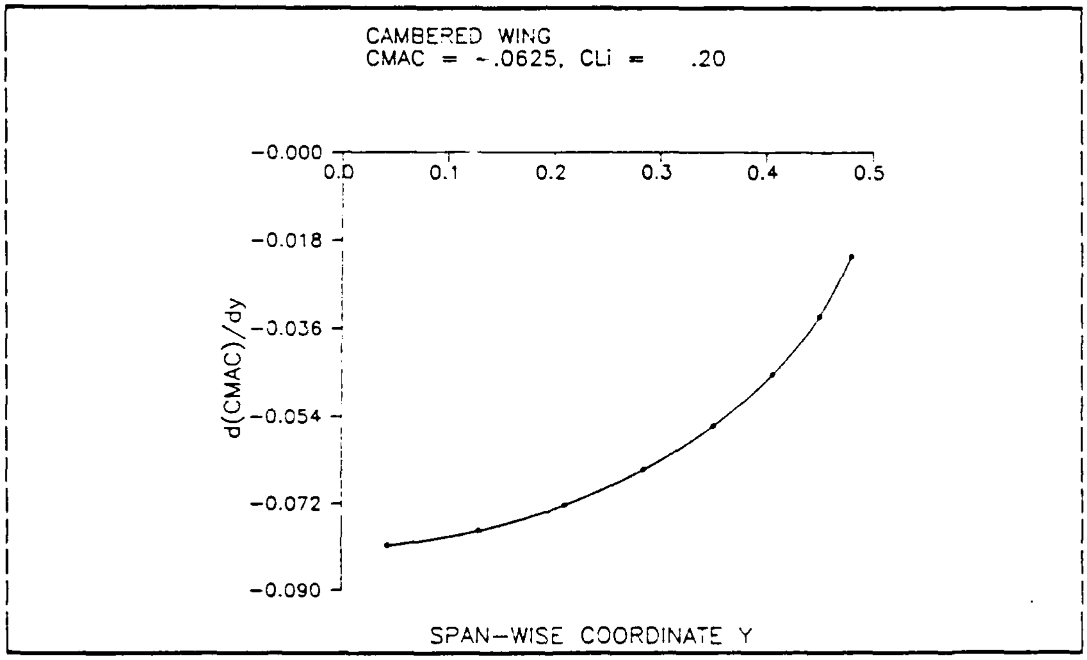
FLAT WING d(CD)/dy vs SPAN
Figure 7-15



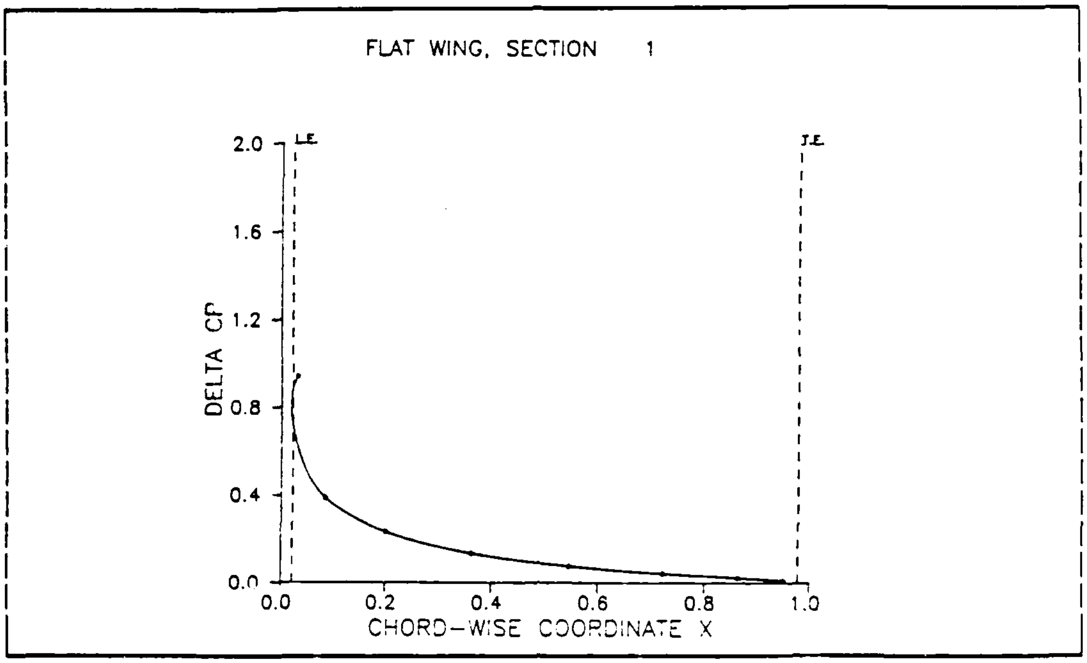
CAMBERED WING d(CD)/dy vs SPAN
Figure 7-16



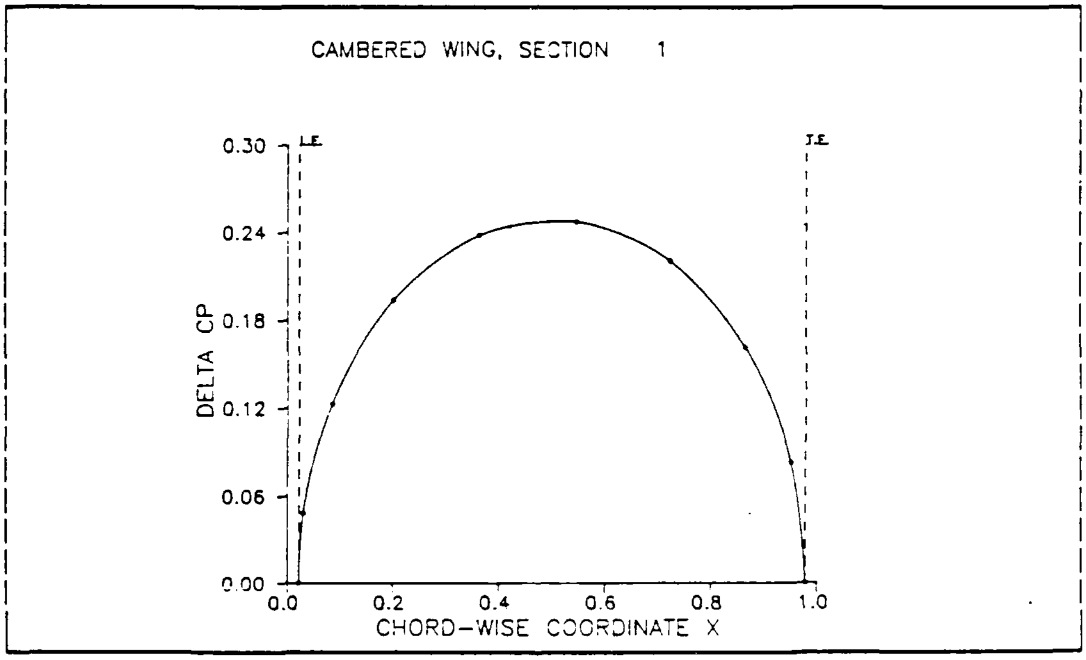
FLAT WING d(CMAC)/dy vs SPAN
Figure 7-17



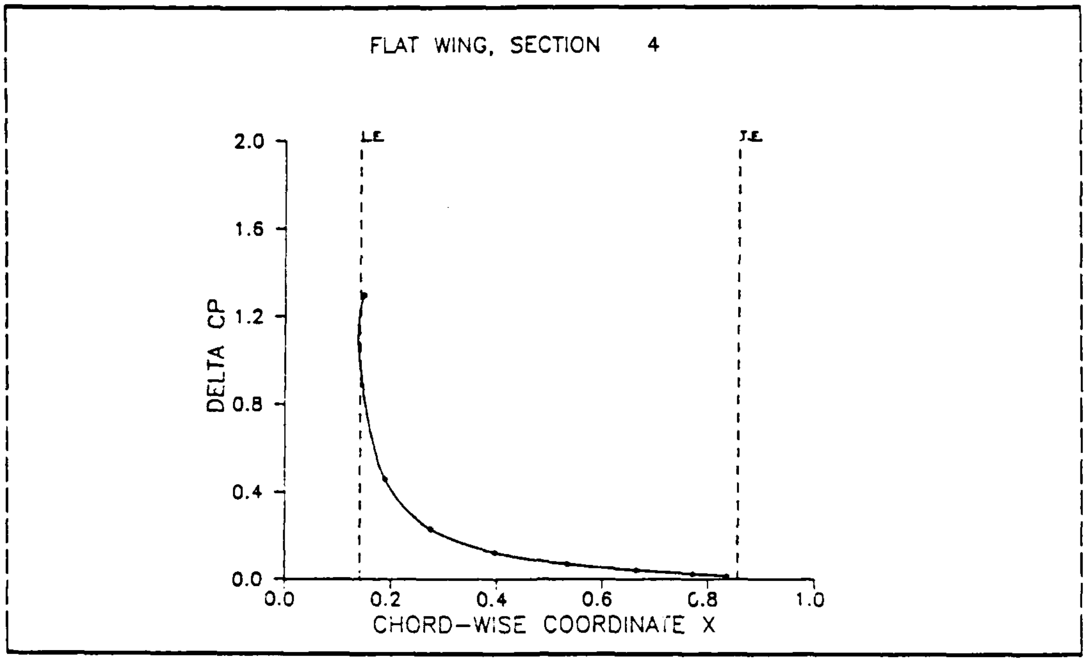
CAMBERED WING d(CMAC)/dy vs SPAN
Figure 7-18



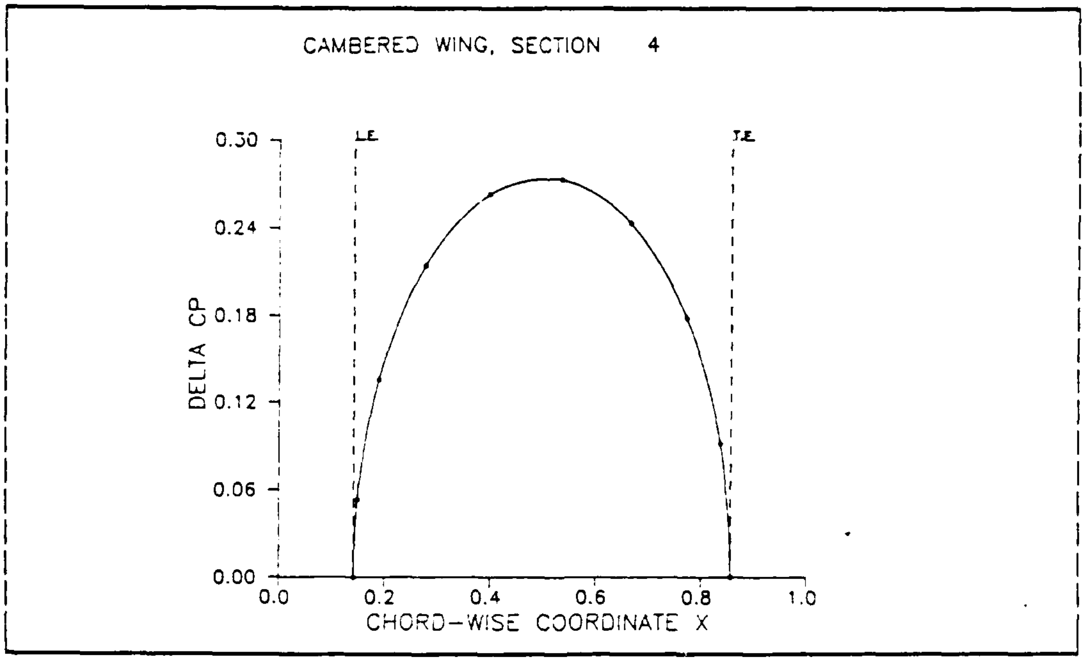
FLAT WING DELTA CP vs CHORD, Section 1
Figure 7-19



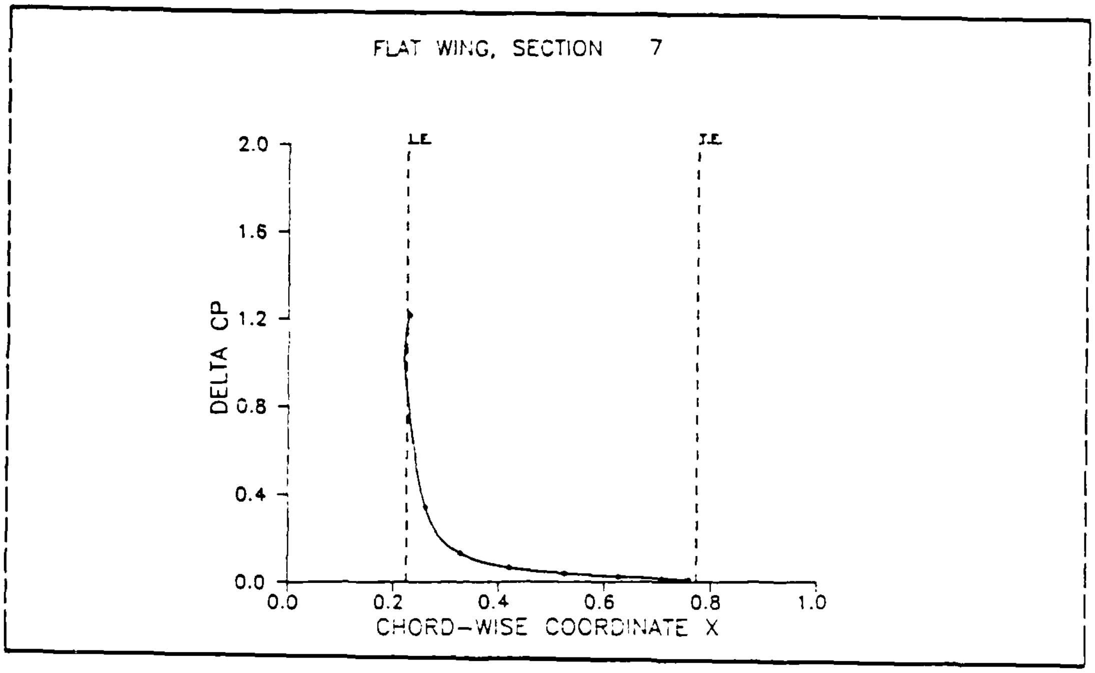
CAMBERED WING DELTA CP vs CHORD, Section 1
Figure 7-20



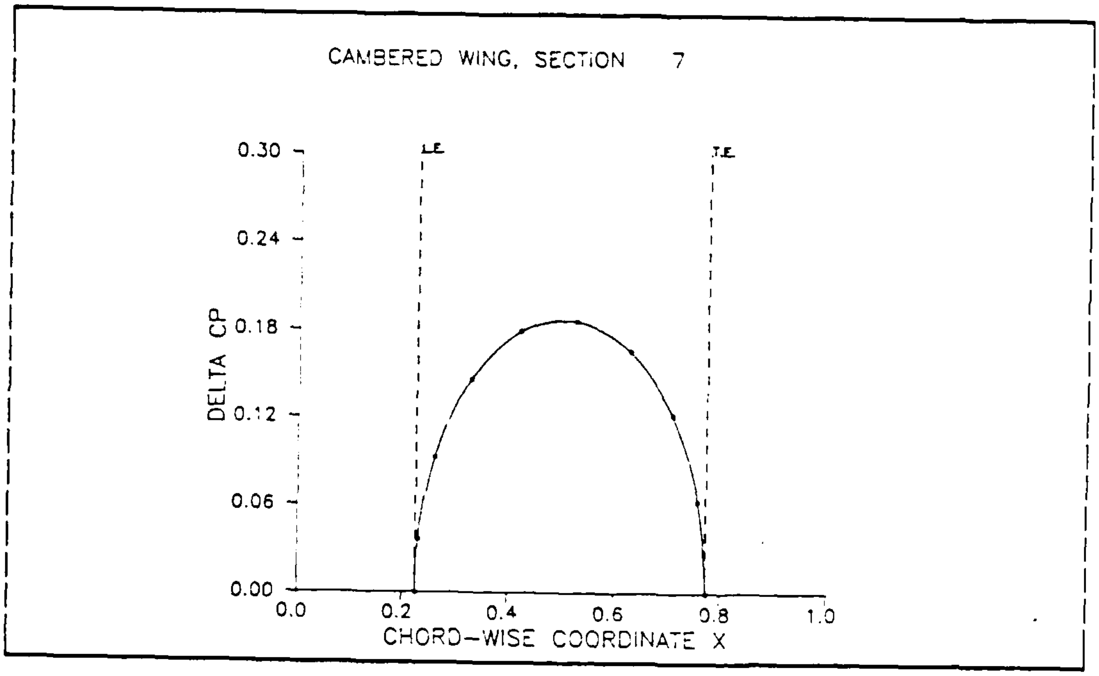
FLAT WING DELTA CP vs CHORD, Section 4
Figure 7-21



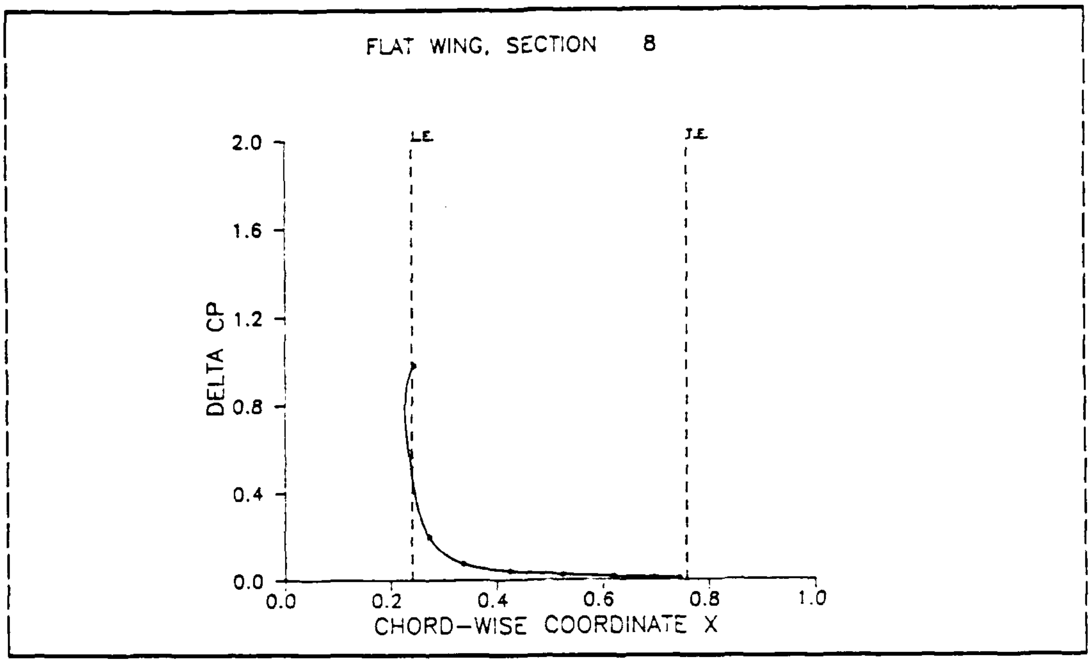
CAMBERED WING DELTA CP vs CHORD, Section 4
Figure 7-22



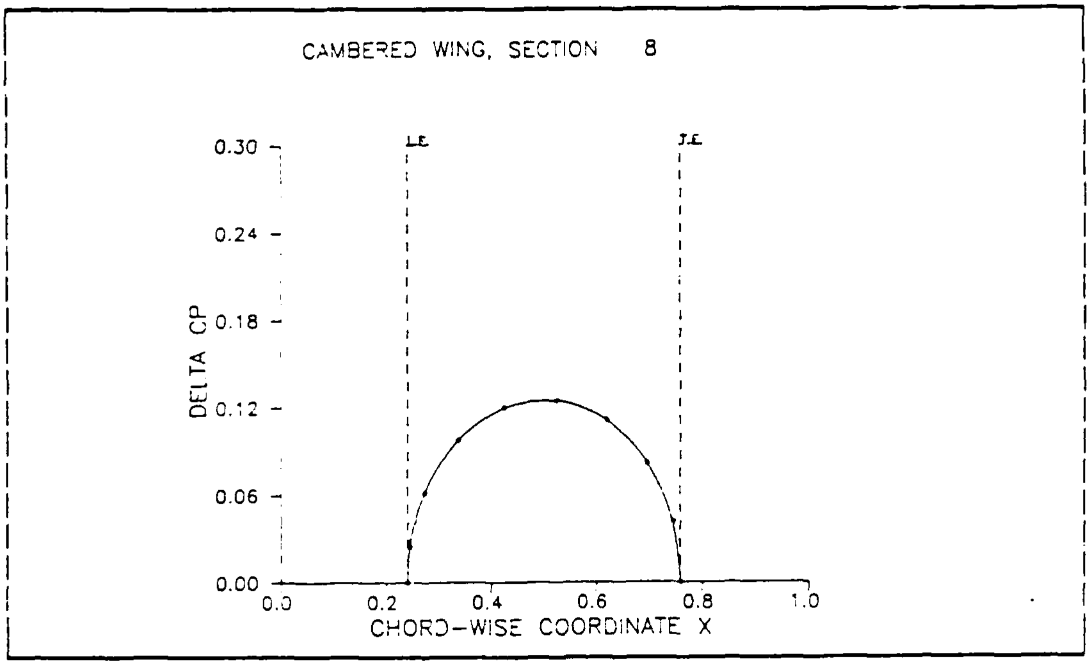
FLAT WING DELTA CP vs CHORD, Section 7
Figure 7-23



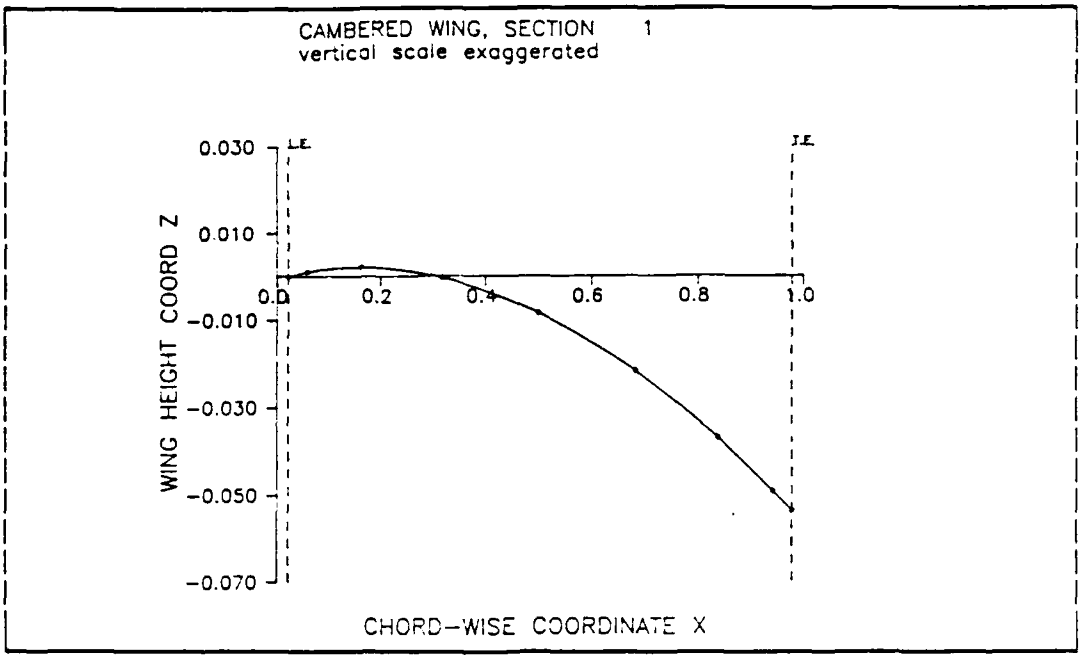
CAMBERED WING DELTA CP vs CHORD, Section 7
Figure 7-24



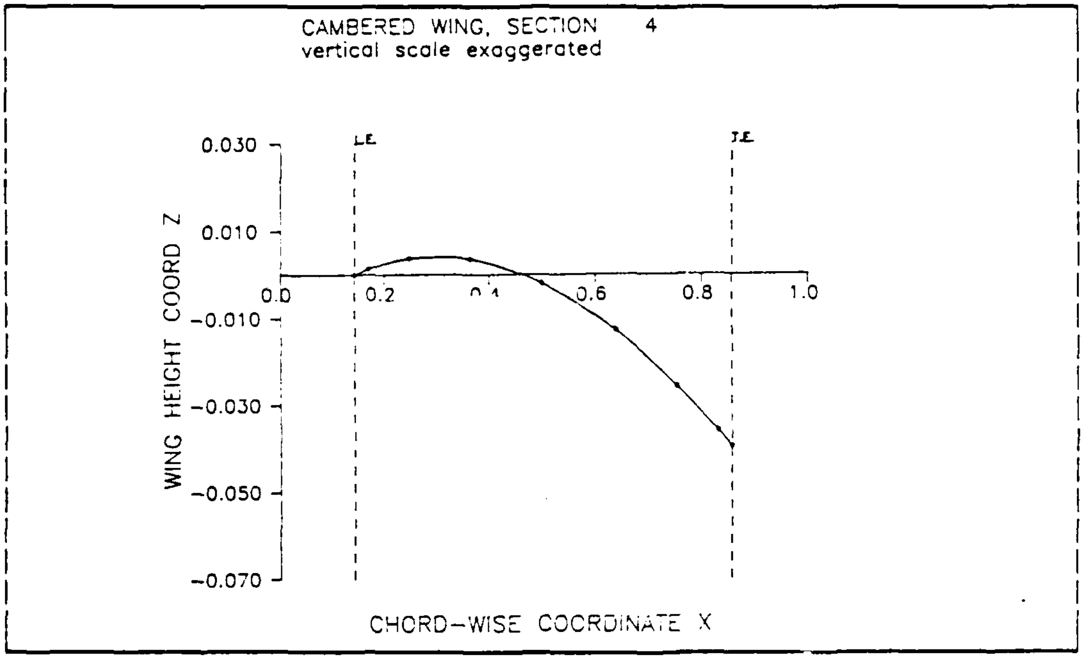
FLAT WING DELTA CP vs CHORD, Section 8
Figure 7-25



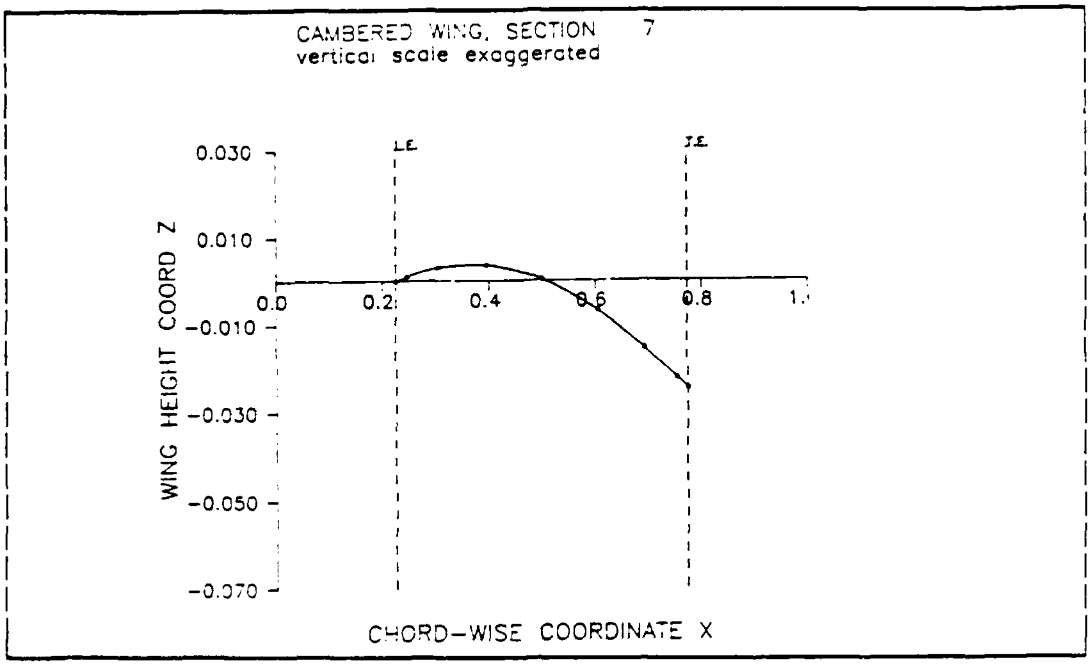
CAMBERED WING DELTA CP vs CHORD, Section 8
Figure 7-26



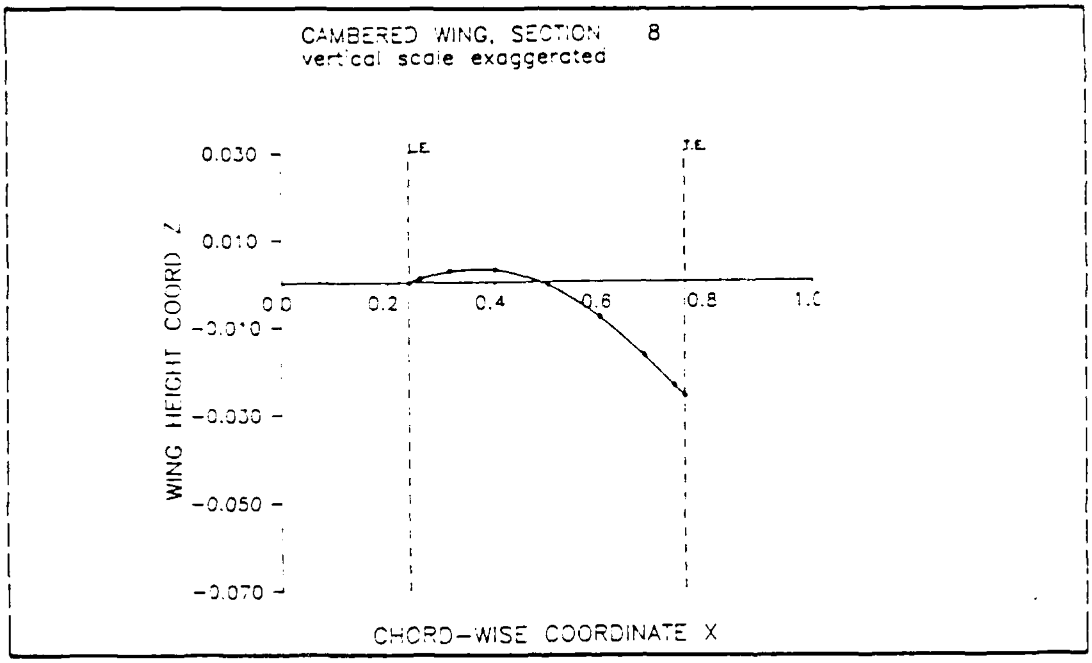
CAMBERED WING SHAPE vs CHORD, Section 1
Figure 7-27



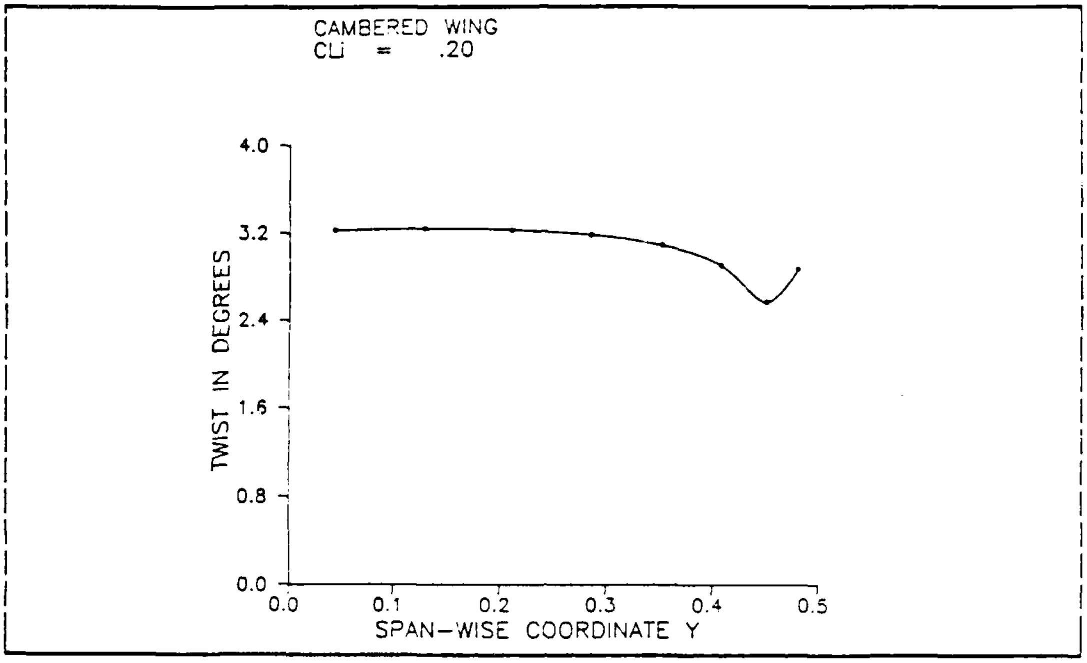
CAMBERED WING SHAPE vs CHORD, Section 4
Figure 7-28



CAMBERED WING SHAPE vs CHORD, Section 7
Figure 7-29



CAMBERED WING SHAPE vs CHORD, Section 8
Figure 7-30



CAMBERED WING TWIST vs SPAN
Figure 7-31

J. METHOD FOR CHANGING LOADING DISTRIBUTION

This is an optional section and may be skipped without loss. Elliptic loading is generated in the SET85 subroutine. The elements which contain the specified loading are WING(IJ,21). If you wish to change the loading from elliptic, realize that the grid elements are not uniform in size.

When generating a ΔC_p function, consider the shape. The load at the wing tips and trailing edge must go to zero. The load at the leading edge should preferably be zero to avoid a singularity at that point.

If you change any of the subroutines, the program must be re-compiled with a FORTRAN compiler to include the changes into the VLAT85.EXE file. The original version of the program was compiled by a MICRO-SOFT 4.2 compiler using the following commands.

```
FL/FPc VLAT85.FOR
```

and

```
FL/FPc GRAP85.FOR
```

If you have a math co-processor, the FPc option allows the program to use the co-processor but does not require one.

K. METHOD FOR CHANGING THE NUMBER OF GRID ELEMENTS

This is also an optional section and can be skipped. In its original form, the program works with up to 100 grid elements in the wing semi-span. The program also has a maximum of 10 chord sections. Fewer grid elements or chord sections will run without modification. If you desire more than 100 grid elements or 10 chord sections, you must modify the program. The changes are not extensive, but require that the programs be recompiled, as described in section J above.

Increase grid size in square increments. Modify the matrix elements as follows. N is the square dimension and corresponds to NX or NY.

For example, if you want 12 wing sections or 12 grid elements in each chord section, make the grid 12x12 (144 elements). In this example, N = 12, and (N*N)+1 = 145. The matrices that must be changed are listed below. The required dimensions are shown.

```
A((N*N)+1,(N*N)+1)

AA((N*N)+1,(N*N)+1)

AS((N*N)+1,(N*N)+1)

WING((N*N)+1,23)

SECTN(N+1,10)
```

These are the line numbers in the programs which contain matrix variables that must be changed.

	Variable				
Program	A()	AA()	AS()	WING()	SECTN()
VLAT85	76	77	77	74	74
LOAD85	56	57	57	54	54
SHAP85	54	55	55	52	52
GAUSS	19	-	-	-	-
MULT1	19	20	20	17	17
MULT2	19	20	20	17	17
MULT3	19	20	20	17	17
GRAP85	52	53	53	50	50
DNWS85	-	-	-	23	23
SET85	-	-	-	69	69

Before increasing the grid density, consider the tradeoff of memory requirement and solution speed. The 10 X 10 grid contains 100 elements and 100 unknowns. A 12 x 12 grid contains 144 elements and 144 unknowns. A 20 x 20 grid contains 400 elements and 400 unknowns. More than 4 times the number of computations are needed to solve a 400 x 400 versus a 100 x 100 matrix. You should also increase precision if you increase the number of grid elements. All elements will be smaller and computer accuracy may be less than the distances between points in the grid.

LIST OF REFERENCES

1. Moran, J., An Introduction to Theoretical and Computational Aerodynamics, pp.130-153, Wiley and Sons, New York, 1984

2. Garner, H.C.; Hewitt, B.L.; and Labrujere, T.E., Comparison of Three Methods for the Evaluation of Subsonic Lifting Surface Theory, Reports and Memoranda 3597, Aeronautical Research Council, London, England, June 1968

3. Hough, G.R., Lattice Arrangements for Rapid Convergence, NASA SP-405, paper presented at the Vortex-Lattice Utilization Workshop, NASA Langley Research Center, Langley, Virginia, May 17-18, 1976

4. Kuethe, A.M. and Chow, C.-Y., Foundations of Aerodynamics, 4th ed., pp.141-152, Wiley and Sons, New York, 1986

5. Barna, P.S., Fluid Mechanics for Engineers, 3rd ed., pp.231-238, Plenum Press, New York, 1969

6. Milne-Thomson, L. M., Theoretical Aerodynamics, 4th ed., pp.216-232, Dover Publications Inc., New York, 1973

APPENDIX A. SOURCE CODE LISTINGS FOR VORTEX PROGRAM

VLAT85

```
***********************************************************************
*                                                                     *
*     PROGRAM VLAT85                                                  *
*                                                                     *
*     THIS IS THE MAIN DRIVER FOR A NUMBER OF SUBROUTINES THAT ARE   *
*     LINKED TOGETHER TO FORM A PROGRAM THAT COMPUTES THE FORCES      *
*     ON A PLANFORM IN INVICID, INCOMPRESSIBLE FLOW.                  *
*     THE SUBROUTINES CALLED ARE LISTED HERE AND AT THE END OF THE    *
*     PROGRAM AS INCLUDE FILES.                                       *
*                                                                     *
*     THIS PROGRAM IS A HIGHLY MODIFIED VERSION OF A PROGRAM IN       *
*     'AN INTRODUCTION TO THEORETICAL AND COMPUTATIONAL AERODYNAMICS' *
*     BY JACK MORAN, WILEY AND SONS, NEW YORK, 1984, FOUND ON PAGE    *
*     151.                                                            *
*                                                                     *
*     PROGRAM VARIABLES ARE:                                          *
*        AR     : ASPECT RATIO                                        *
*        LAMDA  : TAPER RATIO                                         *
*        ADELTA : LEADING EDGE SWEEP ANGLE, IN DEGREES                *
*        DELTA  : LEADING EDGE SWEEP ANGLE, IN RADIANS                *
*        NX     : NUMBER OF GRID ELEMENTS IN A SECTION                *
*        NY     : NUMBER OF GRID ELEMENTS ALONG A SEMI-SPAN           *
*        ALPHA  : ANGLE OF ATTACK, IN DEGREES                      *
*        CLDSRD : DESIRED LIFT COEFFICIENT WITH ELLIPTIC LOADING   *
*        ELLIP  : CHARACTER FLAG TO COMPUTE THE SHAPE FROM AN      *
*                 ELLIPTIC LOADING                                 *
*        PI     : IS PI                                            *
*        NEQNS  : NX*NY, THE TOTAL NUMBER OF EQUATIONS             *
*        A ( )  : A COEFFICIENT MATRIX, USED FOR THE FLOW TANGENCY *
*                 GETS TRANSFORMED IN GAUSS                        *
*        AA ( ) : THE SAME AS A BUT NOT TRANSFORMED IN THE PROGRAM *
*        AS ( ) : ANOTHER COEFFICIENT MATRIX, USED FOR THE DOWN-WASH *
*                 AT THE CENTER OF THE GRID ELEMENT.  REQUIRED TO  *
*                 GET THE FORCES ACTING ON EACH GRID ELEMENT.      *
*        WING( ): A LARGE ARRAY HOLDING VARIOUS IMPORTANT COORDINATES *
*                 FOR EACH GRID ELEMENT, AS WELL AS THE RESULTS FOR *
*                 EACH GRID ELEMENT.  A COMPLETE DESCRIPTION OF EACH *
*                 ELEMENT IN THE ARRAY IS CONTAINED IN THE SUBROUTINE *
*                 SET85                                            *
*        SECTN(): A LARGE ARRAY HOLDING SOME DIMENSIONS FOR THE    *
*                 SECTIONS, AS WELL AS THE COEFFICIENTS FOR EACH   *
*                 SECTION.  A COMPLETE DESCRIPTION OF EACH ELEMENT *
*                 IN THE ARRAY IS CONTAINED IN THE SUBROUTINE SET85. *
*        IIN    : SET TO 5, FOR KEYBOARD INPUT                     *
*        IOUT   : SET TO 6 FOR SCREEN OUTPUT                       *
*        JOUT   : SET TO 10 FOR WING ARRAY INPUT AND OUTPUT        *
*        KOUT   : SET TO 11 FOR SECTN ARRAY INPUT AND OUTPUT       *
*        LOUT   : SET TO 12 FOR LAST PARAMETER INPUT AND OUTPUT    *
*       MOUT   : SET TO 13 FOR FLAT PLATE WING COEFFICIENTS          *
*       NOUT   : SET TO 14 FOR CAMBERED WING COEFFICIENTS            *
*       IMOD   : A FLAG.  IF THE PARAMETERS ARE UNCHANGED FROM THE   *
*                MOST RECENT CALCULATION OF COEFFICIENTS, SET = 0    *
*                IF ANY PARAMETERS ARE CHANGED, SET = 1              *
*       ICHOIC : MENU INPUT VARIABLE                                 *
*       ESC    : THE ESCAPE CHARACTER (27)                           *
*       ITYPE  : INTEGER, 1 IF FLAT, 2 IF ELLIPTIC                   *
*       ISECT  : INTEGER, SECTION NUMBER TO BE PLOTTED.              *
*                                                                    *
*     SUBROUTINES CALLED ARE:                                        *
*       CDAT85 , FORTRAN, GENERAL DATA ENTRY, REAL                   *
*       CDIT85 , FORTRAN, GENERAL DATA ENTRY, INTEGER                *
*       LOAD85 , FORTRAN, COMPUTE LOAD DISTRIBUTION FROM A GIVEN     *
*                WING SHAPE.                                         *
*       SHAP85 , FORTRAN, COMPUTE SHAPE FROM A GIVEN LOAD            *
*                DISTRIBUTION.                                       *
*       DELAY  , FORTRAN, SLOW DOWN THE PROGRAM SO ERROR MESSAGES    *
*                CAN BE SEEN BEFORE BEING ERASED.                    *
*                                                                    *
**********************************************************************
      PROGRAM VLAT85
      REAL LAMDA
      CHARACTER*3 ELLIP
      CHARACTER*1 ESC
      COMMON PI,B,WING(101,23), SECTN(11,10),AR,LAMDA,DELTA,NX,NY,
     * ALPHA,CLDSRD,ELLIP,XAC,CLA
      COMMON/COF/A(101,101), NEQNS
      COMMON/CAM/AA(101,101), AS(101,101)
      COMMON/FILS/IIN,IOUT,JOUT,KOUT,LOUT,MOUT,NOUT
      IMOD =  1
      AR =    2.0
      LAMDA = 1.0
      ADELTA= 10.0
      NX =    4
      NY =    4
      ALPHA = 5.0
      ELLIP = 'YES'
      CLDSRD= 0.5
      ITYPE = 1
      ISECT = 1
      XAC =   0.0
      PI = 2.0*ACOS(0.0)
      ESC = CHAR(27)
************************************************************************
*                                                                      *
*     SET INPUT/OUTPUT VARIABLES AND OPEN THE NECESSARY FILES          *
*     MOUT AND NOUT ARE NOT OPENED UNTIL THE COEFFICIENTS HAVE BEEN    *
*     COMPUTED.                                                        *
*                                                                      *
************************************************************************
      IIN = 5
```

```
          IOUT = 6

          JOUT = 10

          KOUT = 11

          LOUT = 12

          MOUT = 13

          NOUT = 14

          OPEN(UNIT = IIN)

          OPEN(UNIT = IOUT)

          OPEN(UNIT = JOUT, FILE='WING.MAT')

          OPEN(UNIT = KOUT, FILE='SECTION.MAT')

          OPEN(UNIT = LOUT, FILE='PARAMS.DAT')

10        FORMAT (I6/3(F6.2,/),2(I3/),F6.2/A/F6.2/I2/I2/F6.2)

30        FORMAT (23F10.6)

40        FORMAT (10F10.6)

50        FORMAT (' THAT IS NOT A VALID OPTION. ENTER AN OPTION BETWEEN 1'

         * ' AND 11')

60        FORMAT (' ',A,A,F10.6)

62        FORMAT (' ',A,A,I3)

64        FORMAT (' ',A,A,A)

*****************************************************************

*                                                               *

*         READ THE LAST SET OF PARAMETERS, IF UNCHANGED, READ WING AND    *

*         SECTN DATA                                            *

*                                                               *

*****************************************************************

          READ (LOUT,10,ERR=70,END=70) IMOD,AR,LAMDA,ADELTA,NX,NY,ALPHA,
     * ELLIP,CLDSRD,ITYPE,ISECT,XAC
70    DELTA = ADELTA*PI/180.
      B = AR*(1.0+LAMDA)/2.0
      NEQNS = NX*NY
      IF (IMOD.EQ.0) THEN
         READ (JOUT,30) ((WING(I,J),J=1,23),I=1,NEQNS)
         READ (KOUT,40) ((SECTN(I,J).J=1,10),I=1,NY)
      ELSE
      ENDIF
***********************************************************************
*                                                                     *
*     CLEAR THE SCREEN AND WRITE THE MAIN MENU.                       *
*     THIS USES THE ANSI.SYS CONTROL CODES TO CLEAR THE SCREEN AND    *
*     POSITION THE CURSOR.  IT ALSO USES FLASHUP WINDOWS FOR THE      *
*     MENU PROMPTS.                                                   *
*                                                                     *
***********************************************************************
100   WRITE (IOUT,*) ESC,'[2J'
      WRITE (IOUT,*) '~C=ALL/'
      WRITE (IOUT,*) '~W=LOAD/'
      WRITE (IOUT,60) ESC,'[09;69H',AR
      WRITE (IOUT,60) ESC,'[10;69H',LAMDA
      WRITE (IOUT,60) ESC,'[11;69H',ADELTA
      WRITE (IOUT,60) ESC,'[12;69H',ALPHA
      WRITE (IOUT,62) ESC,'[13;69H',NX
      WRITE (IOUT,62) ESC,'[14;69H',NY
      WRITE (IOUT,64) ESC,'[15;69H',ELLIP
      WRITE (IOUT,60) ESC,'[16;69H',CLDSRD
      NEQNS = NX*NY
***********************************************************************
*                                                                     *
*     READ THE OPTION.  IF OUT OF RANGE, PRINT ERROR MSG, AND START   *
*     OVER.                                                           *
*                                                                     *
***********************************************************************
110   READ(IIN,*,END=110,ERR=110) ICHOIC
      WRITE (IOUT,*) ESC,'[2J'
      WRITE (IOUT,*) ESC,'[00;00H'
      IF ((ICHOIC.LT.1).OR.(ICHOIC.GT.11)) THEN
         WRITE (IOUT,50)
         CALL DELAY(6)
         GO TO 100
      ELSE
      ENDIF
***********************************************************************
*                                                                     *
*     MAKE ALL CHANGES TO PARAMETERS AS DESIRED.                      *
*     VALUES PASSED TO CDAT AND CDIT ARE LOW, HIGH, VARIABLE, AND     *
*     NAME OF THE VARIABLE.  IMOD IS RETURNED WITH VALUE 1.           *
*                                                                     *
***********************************************************************
      IF (ICHOIC.EQ.1) CALL CDAT85(0.0,20.0,AR,
     *  'ASPECT RATIO                        ',IMOD)
      IF (ICHOIC.EQ.2) CALL CDAT85(0.0,1.0,LAMDA,
     *  'TAPER RATIO                         ',IMOD)
      IF (ICHOIC.EQ.3) THEN
         CALL CDAT85(0.0,60.0,ADELTA,
     *  'LEADING EDGE SWEEP ANGLE IN DEGREES ',IMOD)
         DELTA = ADELTA*PI/180.
      ELSE
      ENDIF
      IF (ICHOIC.EQ.4) CALL CDAT85(0.0,10.0,ALPHA,
     *  'ANGLE OF ATTACK IN DEGREES          ',IMOD)
      IF (ICHOIC.EQ.5) CALL CDIT85(0,10,NX,
     *  'NUMBER OF ELEMENTS IN A SECTION     ',IMOD)
      IF (ICHOIC.EQ.6) CALL CDIT85(0,10,NY,
     *  'NUMBER OF SECTIONS IN A SEMI-SPAN   ',IMOD)
      IF (ICHOIC.EQ.8) CALL CDAT85(0.0,0.5,CLDSRD,
     *  'DESIRED LIFT COEFFICIENT            ',IMOD)
      IF (ICHOIC.EQ.7) THEN
         IMOD = 1
         IF(ELLIP.EQ.'YES') THEN
              ELLIP = 'NO '
         ELSE
              ELLIP = 'YES'
         ENDIF
      ELSE
      ENDIF
```

```
***********************************************************************
*                                                                     *
*     RUN THE MAIN PART OF THE PROGRAM.  COMPUTE THE LOAD, FROM THE   *
*     FLAT PLATE SHAPE, AND THEN THE SHAPE, FROM THE ELLIPTIC LOAD    *
*     IF THE ELLIP FLAG IS SET.                                       *
*                                                                     *
***********************************************************************
115   IF (ICHOIC.FQ.9) THEN
         IMOD = 0
         WRITE (IOUT,*) ESC,'[2J'
         WRITE (IOUT,*) ESC,'[00;00H'
         WRITE (IOUT,*) '~C=ALL/'
         WRITE (IOUT,*) '~W=WAIT/'
         CALL LOAD85
         IF (ELLIP.EQ.'YES') CALL SHAP85
      ELSE
      ENDIF
***********************************************************************
*                                                                     *
*     OPTION 10, RETURN TO THE MAIN MENU. DO A NORMAL END. WRITE THE  *
*     PARAMETERS AND DATA TO FILES AND CLOSE THEM.                    *
*                                                                     *
***********************************************************************
      IF (ICHOIC.EQ.10) THEN
         REWIND (JOUT)
         REWIND (KOUT)
         REWIND (LOUT)
         WRITE(JOUT,30) ((WING(I,J),J=1,23),I=1,NEQNS)
         WRITE(KOUT,40) ((SECTN(I,J),J=1,10),I=1,NY)
         WRITE(LOUT,10) IMOD,AR,LAMDA,ADELTA,NX,NY,ALPHA,ELLIP,
     *    CLDSRD,ITYPE,ISECT,XAC
         CLOSE (JOUT)
         CLOSE (KOUT)
         CLOSE (LOUT)
         GOTO 1000
      ELSE
      ENDIF
      GO TO 100
1000  WRITE (IOUT,*) ESC,'[2J'
      WRITE (IOUT,*) ESC,'[00;00H'
      WRITE (IOUT,*) '~C=ALL/'
      WRITE (IOUT,*) '~W=MAIN/'
      END
***********************************************************************
*                                                                     *
*     THESE ARE ALL THE SUBROUTINES THAT ARE LINKED,                  *
*                                                                     *
***********************************************************************
$INCLUDE:'SET85.FOR'
$INCLUDE:'DNWS85.FOR'
$INCLUDE:'GAUSS.FOR'
$INCLUDE:'CDAT85.FOR'
```

```
$INCLUDE:'CDIT85.FOR'

$INCLUDE:'LOAD85.FOR'

$INCLUDE:'SHAP85.FOR'

$INCLUDE:'DELAY.FOR'

$INCLUDE:'MULT1.FOR'

$INCLUDE:'MULT2.FOR'

$INCLUDE:'MULT3.FOR'
```

LOAD85

```
*******************************************************************
*                                                                 *
*     SUBROUTINE LOAD85                                           *
*                                                                 *
*     SUBROUTINE TO DEVELOP A LOADING, OR VORTEX DISTRIBUTION, THAT *
*     IS ASSOCIATED WITH A FLAT PLATE WING.  THE LOADING GENERATED *
*     IS RELATED TO THE PRESSURE DIFFERENTIAL, DELTA CP, AND HAS THE *
*     SAME BASIC SHAPE.  THE SOLUTION IS GAINED BY USING A MATRIX  *
*     OF INFLUENCE COEFFICIENTS AND THE WING SHAPE, AND SOLVING FOR *
*     THE VORTEX STRENGTHS.                                        *
*                                                                 *
*     ALL REQUIRED PARAMETERS AND VARIABLES ARE CONTAINED IN COMMON. *
*     A DESCRIPTION OF THE COMMON BLOCK VARIABLES IS CONTAINED IN  *
*     OTHER ROUTINES.                                              *
*                                                                 *
*     VARIABLES USED IN THE SUBROUTINE ARE:                        *
*        ALF    : ANGLE OF ATTACK IN RADIANS                       *
*        DELTA  : THE LEADING EDGE SWEEP ANGLE IN RADIANS          *
*        ADELTA : THE LEADING EDGE SWEEP ANGLE IN DEGREES          *
*        CBAR   : THE AVERAGE CHORD                                *
*        CL     : INTERIM VALUE, USED TO COMPUTE A FINAL VALUE     *
*        CD     : INTERIM VALUE, USED TO COMPUTE A FINAL VALUE     *
*        CM     : INTERIM VALUE, USED TO COMPUTE A FINAL VALUE     *
*        CXJ    : INTERIM VALUE, USED TO COMPUTE A FINAL VALUE     *
*        CZJ    : INTERIM VALUE, USED TO COMPUTE A FINAL VALUE     *
*       CMJ    : INTERIM VALUE, USED TO COMPUTE A FINAL VALUE       *
*       XAC    : X POSITION OF THE AERODYNAMIC CENTER               *
*       CLT    : TOTAL LIFT COEFFICIENT OF THE FLAT WING            *
*       CDT    : TOTAL DRAG COEFFICIENT OF THE FLAT WING            *
*       CMT    : TOTAL MOMENT COEFFICIENT OF THE FLAT WING          *
*                ABOUT THE AERODYNAMIC CENTER                       *
*       CDOCL2 : CD/CL^2, ALSO CALLED K IN THE DRAG POLAR           *
*                                                                   *
*     SUBROUTINES CALLED ARE:                                       *
*       SET85  : FORTRAN, USED TO GENERATE THE WING AND SECTION     *
*                ARRAY CONTAINING PLANFORM COORDINATES.             *
*       DNWS85 : FORTRAN, USED TO COMPUTE THE DOWNWASH GENERATED    *
*                AT A POINT BY A HORSE-SHOE VORTEX AND IT'S MIRROR  *
*                IMAGE ON THE OPPOSITE SEMI-SPAN.                   *
*       GAUSS  : FORTRAN, USED TO SOLVE THE INFLUENCE COEFF MATRIX  *
*                AND SLOPE VECTOR, FOR THE VORTEX STRENGTH.  USES   *
*                SIMPLE GAUSS ELIMINATION.                          *
*       MULT2  : FORTRAN, USED TO COMPUTE THE DOWNWASH GENERATED    *
*                ON EACH VORTEX ELEMENT, NECESSARY TO FIND THE      *
*                FORCES ACTING ON THE WING.                         *
*                                                                   *
*********************************************************************
      SUBROUTINE LOAD85
      REAL LAMDA
      CHARACTER*3, ELLIP
      COMMON PI,B,WING(101,23), SECTN(11,10),AR,LAMDA,DELTA,NX,NY,
     * ALPHA,CLDSRD,ELLIP,XAC,CLA
      COMMON/COF/A(101,101), NEQNS
      COMMON/CAM/AA(101,101), AS(101,101)
      COMMON/FILS/IIN,IOUT,JOUT,KOUT,LOUT,MOUT,NOUT
60    FORMAT ( ' FLAT WING',
     *   /' CL     = ',F12.6/' CD     = ',F12.6/' CD/CL2 = ',F12.6
     *   /' CMAC   = ',F12.6/' XAC    = ',F12.6/' AR     = ',F12.6

     *   /' LAMDA  = ',F12.6/' DELTA  = ',F12.6/' NX     = ',I3
     *   /' NY     = ',I3   /' ALPHA  = ',F12.6/' CLDSRD = ',F12.6

     *   /' ELLIP  = ',A3   /' CLa, Lift Curve Slope'
     *   /' per/Deg= ',F12.6)
************************************************************************
*                                                                      *
*     COMPUTE THE ANGLE OF ATTACK IN RADIANS.                          *
*                                                                      *
************************************************************************
      ALF = ALPHA*PI/180.0
      SINALF = SIN(ALF)
      COSALF = COS(ALF)
************************************************************************
*                                                                      *
*     CALL THE SETUP ROUTINE. USED TO BUILD THE COORDINATES OF THE     *
*     WING.  THEY ARE STORED IN THE ARRAYS WING AND SECTN.             *
*                                                                      *
```

```
***********************************************************************
      CALL SET85
***********************************************************************
*                                                                     *
*     BUILD THE THREE COEFFICIENT MATRICES, A(), AA(), AND AS()       *
*     A() AND AA() ARE THE SAME, AND ARE FOR EVALUATING FLOW          *
*     TANGENCY AT THE EDGE OF THE GRID ELEMENTS.                      *
*     AS() IS FOR EVALUATING DOWN-WASH GENERATED ON THE VORTEX        *
*     ELEMENT ITSELF.                                                   *

*                                                                     *
***********************************************************************
      DO 130 I=1,NY
         DO 120 J=1,NX
              IJ = (I-1)*NX+J
***********************************************************************
*                                                                     *
*     AUGMENT THE A() MATRIX WITH THE RIGHT HAND SIDE, WING SHAPE.    *
*                                                                     *
***********************************************************************
              A(IJ,NEQNS+1) = SINALF
*             A(IJ,NEQNS+1) = ALF
              DO 110 K=1,NY
                   DO 100 L=1,NX
                        KL = (K-1)*NX+L
                        CALL DNWS85(IJ,KL,A(KL,IJ),'CONT')
                        AA(KL,IJ) = A(KL,IJ)
                        CALL DNWS85(IJ,KL,AS(KL,IJ),'SELF')
100                CONTINUE
110           CONTINUE
120      CONTINUE
130   CONTINUE
***********************************************************************
*                                                                     *
*     COMPUTE THE AVERAGE CHORD                                       *
*                                                                     *
***********************************************************************
      CBAR = (SECTN(1,1)+SECTN(NY,1))/2.0
***********************************************************************
*                                                                     *
*     CALL THE GAUSS ROUTINE, TO SOLVE FOR THE LOAD VECTOR            *
*     THE RESULT IS RETURNED IN PLACE OF THE ORIGINAL LOAD VECTOR     *
*                                                                     *
***********************************************************************
      CALL GAUSS(1)
***********************************************************************
*                                                                     *
*     STORE THE RESULT IN THE WING() MATRIX                           *
*     CONVERT THE CIRCULATION, OR VORTEX STRENGTH IN ELEMENT 15 TO   *
*     DELTA CP, IN ELEMENT 16                                         *
*                                                                     *
***********************************************************************
```

```
      DO 210 I = 1,NY
         DO 200 J = 1,NX
              IJ = (I-1)*NX+J
              WING(IJ,15) = A(IJ,NEQNS+1)
              WING(IJ,16) = 2.0*WING(IJ,15)/WING(IJ,13)
200      CONTINUE
210   CONTINUE
************************************************************************
*                                                                      *
*     COMPUTE THE DOWNWASH ON EACH VORTEX ELEMENT, NECESSARY TO        *
*     FIND THE FORCES ACTING ON THE WING.                              *
*                                                                      *
************************************************************************
      CALL MULT2
************************************************************************
*                                                                      *
*     INITIALIZE VALUES AND COMPUTE THE FORCES ACTING ON THE WING      *
*     SECTIONS AND ALSO THE TOTAL WING.                                *
*                                                                      *
************************************************************************
      CD = 0.0
      CL = 0.0
      CM = 0.0
      DO 330 I=1,NY
         CXJ = 0.0
         CZJ = 0.0
         CMJ = 0.0
         DO 320 J=1,NX
              IJ = (I-1)*NX+J
              CZJ =  WING(IJ,15)*2.0*(1.-WING(IJ,22)*SINALF)/(B*CBAR)
              CXJ =  WING(IJ,15)*WING(IJ,22)*2.0*COSALF/(B*CBAR)
              CMJ = -WING(IJ,15)*WING(IJ,4)*2.0*
     *         (1.-WING(IJ,22)*SINALF)/(B*CBAR*CBAR)
              SECTN(I,2) = SECTN(I,2)+CZJ
              SECTN(I,3) = SECTN(I,3)+CXJ
              SECTN(I.4) = SECTN(I,4)+CMJ
320      CONTINUE
         CL = CL+SECTN(I,2)*SECTN(I,6)
         CD = CD+SECTN(I.3)*SECTN(I.6)
         CM = CM+SECTN(I,4)*SECTN(I,6)
         SECTN(I,5) = -SECTN(I,4)/SECTN(I,2)
330   CONTINUE
************************************************************************
*                                                                      *
*     NOW COMPUTE THE MOMENTS ABOUT THF AERODYNAMIC CENTER             *
*                                                                      *
************************************************************************
      XAC = -CM*CBAR/CL
      DO 335 I = 1,NY
         SECTN(I,4) = SECTN(I,4) + SECTN(I,2)*XAC/CBAR
335   CONTINUE
************************************************************************
```

```
*                                                                      *
*     COMPUTE THE FINAL TOTAL WING COEFFICIENTS AND WRITE TO THE       *
*     FILE.                                                            *
*                                                                      *
************************************************************************
      CLT = 2.0*CL
      CDT = 2.0*CD
      CMT = 2.0*CM + (CLT*XAC/CBAR)
      CLA = CLT/ALPHA
      CDOCL2 = CDT/CLT**2
      ADELTA = DELTA * 180./PI
      OPEN (UNIT = MOUT, FILE ='FLAT.DAT')
      WRITE(MOUT,60) CLT,CDT,CDOCL2,CMT,XAC,AR,LAMDA,ADELTA,NX,NY,
     * ALPHA, CLDSRD, ELLIP, CLA
      CLOSE (MOUT)
340   RETURN
      END
```

SHAP85

```
***********************************************************************
*                                                                     *
*     SUBROUTINE SHAP85                                               *
*                                                                     *
*     SUBROUTINE TO DEVELOP A WING SHAPE THAT IS ASSOCIATED WITH      *
*     A SPECIFIED LOADING FUNCTION.  THE LOADING FUNCTION IS USUALLY *
*     ELLIPTIC, IN THE SPAN AND CHORD DIRECTION. THE SHAPE WILL      *
*     USUALLY RESEMBLE A PARABOLIC CAMBER, IN CHORD, AND HAVE SOME    *
*     DEGREE OF TWIST, ALONG THE SPAN.                                *
*                                                                     *
*     ALL REQUIRED PARAMETERS AND VARIABLES ARE CONTAINED IN COMMON.  *
*     A DESCRIPTION OF THE COMMON BLOCK VARIABLES IS CONTAINED IN     *
*     OTHER ROUTINES.                                                 *
*                                                                     *
*     VARIABLES USED IN THE SUBROUTINE ARE:                           *
*        CBAR   : THE AVERAGE CHORD                                   *
*        CLILST : CLREF FROM THE PREVIOUS ITERATION                   *
*        CLREF    : A SCALER, USED TO INCREASE OR DECREASE THE AMPL   *
*                 OF THE SHAPE, IN ORDER TO ACHIEVE THE DESIRED       *
*                 LIFT COEFFICIENT.                                   *
*        MOD    : A COUNTER USED TO LIMIT THE NUMBER OF ITERATIONS    *
*                 THAT THE PROGRAM CAN USE IN AN ATTEMPT TO GENERATE  *
*                 A DESIRED LIFT COEFFICIENT.                         *
*        H      : A TEMPORARY VARIABLE USED WHILE INTEGRATING THE     *
*                 HEIGHT OF THE WING ALONG A CHORD.                   *
*        COSCAM : THE COSINE OF THE ANGLE MADE BY AN INDIVIDUAL GRID  *
*                 ELEMENT AND THE REMOTE VELOCITY.                    *
*        SINCAM : THE SIN OF THE ANGLE MADE BY AN INDIVIDUAL GRID     *
*                 ELEMENT AND THE REMOTE VELOCITY.                    *
*        CCL    : INTERIM VALUE, USED TO COMPUTE A FINAL VALUE        *
*        CCD    : INTERIM VALUE, USED TO COMPUTE A FINAL VALUE        *
*        CCM    : INTERIM VALUE, USED TO COMPUTE A FINAL VALUE        *
*        CXCP   : X POSITION OF THE CENTER OF PRESSURE                *
*        CCLT   : TOTAL LIFT COEFFICIENT OF THE CAMBERED WING.        *
*        CCDT   : TOTAL DRAG COEFFICIENT OF THE CAMBERED WING.        *
*        CCMT   : TOTAL MOMENT COEFFICIENT OF THE CAMBERED WING.      *
*                 ABOUT THE AERODYNAMIC CENTER                        *
*        CCDOCL : CD/CL^2, ALSO CALLED K IN THE DRAG POLAR            *
*                                                                     *
*     SUBROUTINES CALLED ARE:                                         *
*        MULT1  : FORTRAN, USED TO COMPUTE THE SHAPE OF THE WING      *
*                 FROM THE LOADING VECTOR AND THE INFLUENCE COEFF     *
*                 MATRIX                                              *
*        MULT3  : FORTRAN, USED TO COMPUTE THE DOWNWASH GENERATED     *
*                 ON EACH VORTEX ELEMENT, NECESSARY TO FIND THE       *
*                 FORCES ACTING ON THE WING.                          *
*                                                                     *
***********************************************************************
      SUBROUTINE SHAP85
      REAL LAMDA
      CHARACTER*3, ELLIP
      COMMON PI,B,WING(101,23), SECTN(11,10),AR,LAMDA,DELTA,NX,NY,
     * ALPHA,CLDSRD,ELLIP,XAC,CLA
      COMMON/COF/A(101,101), NEQNS
      COMMON/CAM/AA(101,101), AS(101,101)
      COMMON/FILS/IIN,IOUT,JOUT,KOUT,LOUT,MOUT,NOUT
      CLILST = 0.0
      CLREF = 0.05
      MOD = 0
66    FORMAT (' UNABLE TO REACH THAT CLDSRD')
70    FORMAT ( ' CAMBERED WING',
     *   /' CL     = ',F12.6/' CD     = ',F12.6/' CD/CL2 = ',F12.6
     *   /' CMAC   = ',F12.6/                    ' AR     = ',F12.6
     *   /' LAMDA  = ',F12.6/' DELTA  = ',F12.6/' NX     = ',I3
     *   /' NY     = ',I3   /' ALPHA  = ',F12.6/' CLDSRD = ',F12.6
     *   /' ELLIP  = ',A3)
**********************************************************************
*                                                                    *
*     COMPUTE THE AVERAGE CHORD                                      *
*                                                                    *
**********************************************************************
      CBAR = (SECTN(1,1)+SECTN(NY,1))/2.0
**********************************************************************
*                                                                    *
*     USE THE SCALER, CLREF, TO PUT AN ELLIPTIC LOAD DISTRIBUTION      *
*     IN THE VECTOR (IJ,17).  AT THE SAME TIME COMPUTE THE DELTA CP  *
*     IN THE VECTOR (IJ,18).                                         *
*                                                                      *
************************************************************************
390   DO 400 IJ = 1,NEQNS
         WING(IJ,17) = WING(IJ,21)*CLREF
         WING(IJ,18) = 2.0*WING(IJ,17)/WING(IJ,13)
400   CONTINUE
************************************************************************
*                                                                      *
*     CALL THE SUBROUTINE MULT1 WHICH WILL                             *
*     MULTIPLY THE LOAD VECTOR, (IJ,17) BY THE COEFFICIENT MATRIX,     *
*     AA( ), TO GET THE SHAPE VECTOR, (IJ,19)                          *
*                                                                      *
************************************************************************
      CALL MULT1
************************************************************************
*                                                                      *
*     CHECK THE RESULTING SHAPE VECTOR (IJ,19), TO SEE IF THE ANGLE    *
*     THAT WAS REQUIRED HAS A SIN GREATER THAN 1, THAT IS SIMPLY       *
*     CHECKING THE SHAPE VECTOR, TO SEE IF IT IS GREATER THAN 1.       *
*     IF IT IS, THE SCALER MULTIPLIER, CLREF, IS DECREASED.  AT THE      *
*     SAME TIME, MOD, THE ITERATION LIMITER, IS INCREMENTED.  IF       *
*     MOD IS GREATER THAN 5, THAT MEANS WE OVERSHOT 5 TIMES, AND       *
*     PROBABLY WILL NEVER GET TO THE DESIRED LIFT COEFFICIENT.         *
*     IN THAT CASE, COMPUTE THE LAST VALID SHAPE  AND PROVIDE THOSE    *
*     DATA.                                                            *
*                                                                      *
```

```
************************************************************************
      DO 410 IJ = 1,NEQNS
         IF (ABS(WING(IJ,19)).GT.1.0) THEN
              CLREF = CLREF*.9
              MOD = MOD + 1
              IF (MOD.GT.5) THEN
                 WRITE (IOUT,66)
                 CALL DELAY(6)
                 GO TO 540
              ELSE
              GO TO 390
              ENDIF
         ELSE
         ENDIF
410   CONTINUE
************************************************************************
*                                                                      *
*     BEGIN TO COMPUTE THE FORCE AND MOMENT COEFFICIENTS ON THE        *
*     CAMBERED WING.  MULT3 WILL MULTIPLY THE SPECIFIED LOADING        *
*     BY THE COEFFICIENT MATRIX, AS, TO GET THE DOWNWASH AT THE        *
*     CENTER OF EACH CAMBERED ELEMENT.  INTERMEDIATE VARIABLES ARE     *
*     INITIALIZED.                                                     *
*     COEFFICIENTS FOR SECTIONS AND THE WHOLE WING ARE COMPUTED.       *
*                                                                      *
************************************************************************
      CALL MULT3
      CCD = 0.0
      CCL = 0.0
      CCM = 0.0
      DO 530 I=1,NY
         CCXJ = 0.0
         CCZJ = 0.0
         CCMJ = 0.0
         SECTN(I,7) = 0.0
         SECTN(I,8) = 0.0
         SECTN(I,9) = 0.0
         H = 0.0
         IJ = (I-1)*NX+1
         DO 520 J=1,NX
              IJ = (I-1)*NX+J
              H = H-WING(IJ,19)*WING(IJ,13)
              WING(IJ,20) = H
              COSCAM = COS(ASIN(WING(IJ,19)))
              SINCAM = WING(IJ,19)
              CCZJ = WING(IJ,17)*2.0*(1.-WING(IJ,23)*SINCAM)/(B*CBAR)
              CCXJ = WING(IJ,17)*WING(IJ,23)*2.0*COSCAM/(B*CBAR)
              CCMJ = -WING(IJ,17)*WING(IJ,4)*2.0*
     *                   (1.-WING(IJ,23)*SINCAM)/(B*CBAR*CBAR)
              SECTN(I,7) = SECTN(I,7)+CCZJ
              SECTN(I,8) = SECTN(I,8)+CCXJ
              SECTN(I,9) = SECTN(I,9)+CCMJ
520      CONTINUE
```

```
         CCL = CCL+SECTN(I,7)*SECTN(I,6)
         CCD = CCD+SECTN(I,8)*SECTN(I,6)
         CCM = CCM+SECTN(I,9)*SECTN(I,6)
************************************************************************
*                                                                      *
*     COMPUTE THE SECTION TWIST.  THE CAMBERED WING CL                 *
*     WILL BE THE SAME AS CLDSRD, WITHIN A VERY SMALL MARGIN.         *
*                                                                      *
************************************************************************
         SECTN(I,10) = -WING(I*NX,20)*180./(SECTN(I,1)*PI)
************************************************************************
*                                                                      *
*     COMPUTE THE MOMENT COEFFICIENT ABOUT THE AERODYNAMIC CENTER.     *
*                                                                      *
************************************************************************
         SECTN(I,9) = SECTN(I,9) + (SECTN(I,7)*XAC/CBAR)
530   CONTINUE
      CXCP = -CCM/CCL
      CCLT = 2.0*CCL
      CCDT = 2.0*CCD
      CCMT = 2.0*CCM + (CCLT*XAC/CBAR)
      CCDOCL = CCDT/CCLT**2
      CLILST = CLREF
      IF (ABS(CLDSRD-CCLT).LT.(CLDSRD*0.01)) GO TO 540
      CLREF = CLREF*CLDSRD/CCLT
      GO TO 390
```

```
***********************************************************************
*                                                                     *
*     THE DESIRED LIFT COEFFICIENT HAS BEEN ACHIEVED, WITHIN A        *
*     A TOLERANCE OF +/- 1.0%. THE CORRECT VORTEX STRENGTH AND        *
*     DELTA CP ARE PUT INTO THE WING ARRAY, AND THE FINAL CAMBERED    *
*     WING VALUED OF LIFT, DRAG, K, MOMENT, AND CENTER OF PRESSURE    *
*     ARE PRINTED.                                                    *
*                                                                     *
***********************************************************************
540   CONTINUE
      DO 550 IJ = 1,NEQNS
         WING(IJ,17) = WING(IJ,21)*CLILST
         WING(IJ,18) = 2.0*WING(IJ,17)/WING(IJ,13)
550   CONTINUE
***********************************************************************
*                                                                     *
*     WRITE THE TOTAL WING COEFFICIENTS TO THE SCREEN                 *
*                                                                     *
***********************************************************************
      OPEN (UNIT = MOUT, FILE = 'CAMBER.DAT')
      ADELTA = DELTA *180./PI
      WRITE(MOUT,70) CCLT,CCDT,CCDOCL,CCMT,     AR,LAMDA,ADELTA,NX,NY,
     * ALPHA, CLDSRD, ELLIP
      CLOSE (MOUT)
      RETURN
      END
```

SET85

```
*************************************************************************
*                                                                       *
*     SUBROUTINE SET85                                                  *
*                                                                       *
*     This program is written for the setup of a straighthorse shoe     *
*     vortex over  tapered and swept wing. It uses cosine spacing       *
*     for both the span and chord elements.  This concentrates          *
*     elements near the edges of the wing, where they are most          *
*     important.                                                        *
*                                                                       *
*     Column elements in the WING array are:                            *
*     1  sequential position, also called IJ                            *
*     2  integer y position                                             *
*     3  integer x position                                             *
*     4  x center of element                                            *
*     5  y center of element                                            *
*     6  x  position of horse shoe vortex                               *
*     7  blank                                                          *
*     8  ya position of one corner of horse shoe vortex                 *
*     9  yb position of one corner of horse shoe vortex                 *
*     10 xp position of flow tangency point, 3/4 chord                  *
*     11 xp position of down wash point, 1/4 chord                      *
*     12 yp position of flow tangency or down wash pt, center of elem   *
*     13 del x, chord wise dimension                                    *
*     14 del y, span wise dimension                                     *
*     15 flat plate vorticity                                           *
*     16 flat plate delta cp                                            *
*     17 cambered voricity, adjusted for CLi                            *
*     18 cambered delta cp                                              *
*     19 cambered slope                                                 *
*     20 cambered height                                                *
*     21 cambered vorticity, if CLREF were = 1.0                        *
*     22 downwash at center of flat plate element                       *
*     23 downwash at center of cambered element                         *
*                                                                       *
*     Column elements in the SECTN array are:                           *
*     1  chord                                                          *
*     2  flat wing lift force per unit span                             *
*     3  flat wing drag force per unit span                             *
*     4  flat wing moment about the aerodynamic center                  *
*     5  flat wing XCP, center of pressure relative to x=0              *
*     6  section delta y                                                *
*     7  cambered wing lift force per unit span                         *
*     8  cambered wing drag force per unit span                         *
*     9  cambered wing moment about the aerodynamic center              *
*     10 cambered wing twist.                                           *
*                                                                       *
*     VARIABLES USED ARE:                                               *
*        B      :  WING SPAN                                            *
*        DTHE   :  AN ANGULAR ELEMENT WIDTH, IN RADIANS  SPAN           *
*        DPSI   :  AN ANGULAR ELEMENT DEPTH, IN RADIANS, CHORD          *
*       THEA   :  INTERMEDIATE VARIABLE                                 *
*       THEA   :  INTERMEDIATE VARIABLE                                 *
*       PSIA   :  INTERMEDIATE VARIABLE                                 *
*       PSIB   :  INTERMEDIATE VARIABLE                                 *
*       ELE    :  INTERMEDIATE VARIABLE                                 *
*       ELT    :  INTERMEDIATE VARIABLE                                 *
*       PSI    :  CHORDWISE ANGULAR COORD FOR THE CENTER OF AN          *
*                 ELEMENT.  USED IN GENERATING ELLIPTIC LOAD.           *
*       THETA  :  SPANWISE ANGULAR COORD FOR THE CENTER  OF AN          *
*                 ELEMENT. USED IN GENERATING ELLIPTIC LOAD.            *
*                                                                       *
*     FUNCTIONS USED ARE:
*       BETA   :  COMPUTES THE LENGTH OF A LOCAL SECTION CHORD.         *
*                                                                       *
*     ALL VARIABLES AND PARAMETERS ARE PASSED IN COMMON BLOCKS          *
*     THE ONLY VARIABLES CHANGED IN THIS ROUTINE ARE:                   *
*       B                                                               *
*       WING()                                                          *
*       SECTN()                                                         *
*                                                                       *
*************************************************************************
      SUBROUTINE SET85
      COMMON PI,B,WING(101,23), SECTN(11,10),AR,LAMDA,DELTA,NX,NY,
     * ALPHA,CLDSRD,ELLIP,XAC,CLA
      REAL LAMDA
*************************************************************************
```

```
*                                                                      *
*     DEFINE THE FUNCTION BETA, WHICH COMPUTES THE LENGTH OF A LOCAL   *
*     CHORD.                                                           *
*                                                                      *
************************************************************************
      BETA(Y,LAMDA,AR) = 1.0-(Y*4.0*(1.0-LAMDA)/(AR*(1.0+LAMDA)))
************************************************************************
*                                                                      *
*     DEFINE THE SIZE OF SPAN AND CHORD WISE GRID ELEMENTS.            *
*     DETERMINE THE TOTAL WING SPAN.                                   *
*                                                                      *
************************************************************************
*     DTHE = PI*0.5/FLOAT(NY)
************************************************************************
*                                                                      *
*     THIS STATEMENT SETS THE GRID UP SO THAT THE OUTER ELEMENT        *
*     IS IGNORED.                                                      *
*                                                                      *
************************************************************************
      DTHE = PI*0.5/FLOAT(NY+1)
      DPSI = PI/FLOAT(NX)
      B = AR*(1.0+LAMDA)/2.0
      DO 110 I = 1,NY
         DO 100 J = 1,NX
************************************************************************
*                                                                      *
*     COMPUTE ALL REQUIRED COORDINATES FOR THE WING() ARRAY.                *
*                                                                           *
*****************************************************************************
              IJ = (I-1)*NX+J
              WING(IJ,1) = FLOAT(IJ)
              WING(IJ,2) = FLOAT(I)
              WING(IJ,3) = FLOAT(J)
              THEA = (FLOAT(I-1)*DTHE)+(PI/2.0)
              THEB = (FLOAT(I)*DTHE)+(PI/2.0)
              PSIA = (FLOAT(J-1)*DPSI)
              PSIB = (FLOAT(J)*DPSI)
              WING(IJ,8) = -B*COS(THEA)/2.0
              WING(IJ,9) = -B*COS(THEB)/2.0
              WING(IJ,12) = -B*(COS(THEA)+COS(THEB))/4.0
              ELE = (1.0 - COS(PSIA))*
     *             BETA(WING(IJ,12),LAMDA,AR)/2.0
              ELT = (1.0 - COS(PSIB))*
     *             BETA(WING(IJ,12),LAMDA,AR)/2.0
*     THESE TWO STATEMENTS SET THE POINTS AT 1/4 AND 3/4 CHORD
              WING(IJ,10) = WING(IJ,12)*TAN(DELTA)+(ELE+3*ELT)/4.0
              WING(IJ,11) = WING(IJ,12)*TAN(DELTA)+(3*ELE+ELT)/4.0
              WING(IJ,4) = WING(IJ,12)*TAN(DELTA)+(ELE+ELT)/2.0
              WING(IJ,6) = WING(IJ,11)
              WING(IJ,5) = WING(IJ,12)
              WING(IJ,13) = ELT-ELE
              WING(IJ,14) = WING(IJ,9)-wING(IJ,8)
              PSI = ACOS(1.0-(2.0*(WING(IJ,11)-(WING(IJ,5)*
     *          TAN(DELTA)))/BETA(WING(IJ,5),LAMDA,AR)))
              THETA = ACOS(-2.0*WING(IJ,5)/B)
              WING(IJ,21) = 4.0*(1.0+LAMDA)*SIN(THETA)*SIN(PSI)*
     *          WING(IJ,13)/(PI**2*BETA(WING(IJ,5),LAMDA,AR))
              WING(IJ,7) = 0.0
              WING(IJ,15) = 0.0
              WING(IJ,16) = 0.0
              WING(IJ,17) = 0.0
              WING(IJ,18) = 0.0
              WING(IJ,19) = 0.0
              WING(IJ,20) = 0.0
              WING(IJ,22) = 0.0
              WING(IJ,23) = 0.0
100      CONTINUE
************************************************************************
*                                                                      *
*     COMPUTE ALL REQUIRED COORDINATES FOR THE SECTN() ARRAY.          *
*                                                                      *
************************************************************************
         SECTN(I,1) = BETA(WING(IJ,12),LAMDA,AR)
         SECTN(I,2) = 0.0
         SECTN(I,3) = 0.0
         SECTN(I,4) = 0.0
         SECTN(I,5) = 0.0
         SECTN(I,6) = WING(IJ,14)
         SECTN(I,7) = 0.0
         SECTN(I,8) = 0.0
         SECTN(I,9) = 0.0
         SECTN(I,10) = 0.0
110   CONTINUE
      RETURN
      END
```

DNWS85

```
**********************************************************************
*                                                                    *
*    SUBROUTINE DNWS85(IJ,KL,W,IND)                                  *
*                                                                    *
*    COMPUTE DOWNWASH ON GRID ELEMENT NUMBER KL DUE TO               *
*    VORTICES ON GRID ELEMENT IJ, AND IT'S MIRROR IMAGE              *
*                                                                    *
*    VARIABLES USED ARE:                                             *
*       IJ      :  VORTEX GRID ELEMENT NUMBER, INPUT                 *
*       KL      :  DOWNWASH GRID ELEMENT NUMBER, INPUT               *
*       W       :  THE DOWN WASH, RETURNED TO THE CALLING PROGRAM    *
*                    OUTPUT                                             *

*       IND     :  A FLAG TO DETERMINE WHETHER TO USE TRAILING       *
*                  EDGE OF THE GRID ELEMENT, OR THE CENTER OF        *
*                  THE ELEMENT, INPUT                                *
*                                                                    *
*    FUNCTIONS USED ARE:                                             *
*       WHV1    :  FORTRAN, COMPUTES THE DOWNWASH DUE TO ONE CORNER  *
*                  OF THE HORSESHOE VORTEX                           *
*       WHV2    :  FORTRAN, COMPUTES THE DOWNWASH DUE TO JUST THE    *
*                  TRAILING FILAMENTS OF THE HORSESHOE VORTEX        *
*                                                                    *
**********************************************************************
      SUBROUTINE DNWS85(IJ,KL,W,IND)
      COMMON PI,B,WING(101,23), SECTN(11,10),AR,LAMDA,DELTA,NX,NY,
     * ALPHA,CLDSRD,ELLIP,XAC,CLA
      CHARACTER*4 IND
      IF(IND.EQ.'CONT') GOTO 100
      IF(IND.EQ.'SELF') GOTO 110
100   W = WHV1(WING(KL,10),WING(KL,12),WING(IJ,6),WING(IJ,8))
     *  - WHV1(WING(KL,10),WING(KL,12),WING(IJ,6),WING(IJ,9))
     *  - WHV1(WING(KL,10),WING(KL,12),WING(IJ,6),-WING(IJ,8))
     *  + WHV1(WING(KL,10),WING(KL,12),WING(IJ,6),-WING(IJ,9))
      W = W*.25/PI
      RETURN
110   W = WHV2(WING(KL,11),WING(KL,12),WING(IJ,6),WING(IJ,8))
     *  - WHV2(WING(KL,11),WING(KL,12),WING(IJ,6),WING(IJ,9))
     *  - WHV2(WING(KL,11),WING(KL,12),WING(IJ,6),-WING(IJ,8))
     *  + WHV2(WING(KL,11),WING(KL,12),WING(IJ,6),-WING(IJ,9))
120   W = W*.25/PI
      RETURN
      END
************************************************************************
*                                                                      *
*     FUNCTION WHV1(X1,Y1,X2,Y2)                                       *
*                                                                      *
************************************************************************
      FUNCTION WHV1(X1,Y1,X2,Y2)
      IF (ABS(X1-X2).LT..0001) GOTO 100
      WHV1 = (1.0+SQRT((X1-X2)**2 + (Y1-Y2)**2)/(X1-X2))/(Y1-Y2)
      RETURN
100   WHV1 = 1.0/(Y1-Y2)
      RETURN
      END
************************************************************************
*                                                                      *
*     FUNCTION WHV2(X1,Y1,X2,Y2)                                       *
*                                                                      *
************************************************************************
      FUNCTION WHV2(X1,Y1,X2,Y2)
      IF (ABS(X1-X2).LT..0001) GOTO 100
      WHV2 = (1.0+(X1-X2)/SQRT((X1-X2)**2 + (Y1-Y2)**2))/(Y1-Y2)
      RETURN
100   WHV2 = 1.0/(Y1-Y2)
      RETURN
      END
```

MULT1

```
*********************************************************************
*                                                                   *
*     SUBROUTINE MULT1                                              *
*                                                                   *
*     SUBROUTINE TO MULTIPLY THE SPECIFIED LOADING VECTOR, (I,17)    *
*     BY THE COEFFICIENT MATRIX, AA( ), TO GET THE WING SHAPE       *
*     VECTOR (I,19).                                                *
*                                                                   *
*     VARIABLE NAMES ARE CONTAINED IN THE MAIN PROGRAM.             *
*     ALL VARIABLES AND PARAMETERS ARE PASSED IN COMMON.            *
*     THE ONLY VARIABLE CHANGED IS WING(I,19)                       *
*                                                                   *
*********************************************************************
      SUBROUTINE MULT1
      REAL LAMDA
      CHARACTER*3 ELLIP
      COMMON PI,B,WING(101,23), SECTN(11,10),AR,LAMDA,DELTA,NX,NY,
     * ALPHA,CLDSRD,ELLIP,XAC,CLA
      COMMON/COF/A(101,101), NEQNS
      COMMON/CAM/AA(101,101), AS(101,101)
      DO 110 I = 1,NEQNS
         WING(I,19) = 0.0
         DO 100 J = 1,NEQNS
              WING(I,19) = WING(I,19)+AA(I,J)*WING(J,17)
100      CONTINUE
```

```
110   CONTINUE
      RETURN
      END
```

MULT2

```
***********************************************************************
*                                                                     *
*     SUBROUTINE MULT2                                                *
*                                                                     *
*     SUBROUTINE TO MULTIPLY THE FLAT PLATE LOAD VECTOR, (I,15)       *
*     BY THE COEFFICIENT MATRIX, AS( ), TO GET THE DOWN WASH          *
*     VECTOR (I,22).                                                  *
*                                                                     *
*     VARIABLE NAMES ARE CONTAINED IN THE MAIN PROGRAM.               *
*     ALL VARIABLES AND PARAMETERS ARE PASSED IN COMMON.              *
*     THE ONLY VARIABLE CHANGED IS WING(I,22)                         *
*                                                                     *
***********************************************************************
      SUBROUTINE MULT2
      REAL LAMDA
      CHARACTER*3,ELLIP
      COMMON PI,B,WING(101,23), SECTN(11,10),AR,LAMDA,DELTA,NX,NY,
     * ALPHA,CLDSRD,ELLIP,XAC,CLA
      COMMON/COF/A(101,101), NEQNS
      COMMON/CAM/AA(101,101), AS(101,101)
      DO 110 I = 1,NEQNS
         WING(I,22) = 0.0
         DO 100 J = 1,NEQNS
              WING(I,22) = WING(I,22)+AS(I,J)*WING(J,15)
100      CONTINUE
```

```
110   CONTINUE
      RETURN
      END
```

MULT3

```
***********************************************************************
*                                                                     *
*     SUBROUTINE MULT3                                                *
*                                                                     *
*     SUBROUTINE TO MULTIPLY THE SPECIFIED LOADING VECTOR, (I,17)     *
*     BY THE COEFFICIENT MATRIX, AS( ), TO GET THE DOWN WASH          *
*     VECTOR (I,23).                                                  *
*                                                                     *
*     VARIABLE NAMES ARE CONTAINED IN THE MAIN PROGRAM.               *
*     ALL VARIABLES AND PARAMETERS ARE PASSED IN COMMON.              *
*     THE ONLY VARIABLE CHANGED IS WING(I,23)                         *
*                                                                     *
***********************************************************************
      SUBROUTINE MULT3
      REAL LAMDA
      CHARACTER*3 ELLIP
      COMMON PI,B,WING(101,23), SECTN(11,10),AR,LAMDA,DELTA,NX,NY,
     * ALPHA,CLDSRD,ELLIP,XAC,CLA
      COMMON/COF/A(101,101), NEQNS
      COMMON/CAM/AA(101,101), AS(101,101)
      DO 110 I = 1,NEQNS
         WING(I,23) = 0.0
         DO 100 J = 1,NEQNS
              WING(I,23) = WING(I,23)+AS(I,J)*WING(J,17)
100      CONTINUE
```

```
110   CONTINUE
      RETURN
      END
```

GRAP85

```
***********************************************************************
*                                                                     *
*     PROGRAM GRAP85                                                  *
*                                                                     *
*     THIS IS A PROGRAM WHICH GENERATES A MENU OF GRAPHIC OPTIONS     *
*     AND WRITES FILES TO DISK WHICH ARE USED BY GRAPHER, A SOFTWARE  *
*     PROGRAM FROM GOLDEN SOFTWARE, INC. BOX 281, GOLDEN, COLO.,      *
*     80402.  IF YOU DON'T HAVE A COPY OF THAT SOFTWARE, YOU MUST     *
*     USE SOME OTHER METHOD OF PRESENTING THE DATA.                   *
*                                                                     *
*     DATA FILES USED FOR GRAPHIC OUTPUT ARE:                         *
*     30 = X,Y DATA FILE                                              *
*     32 = GRAPH FILE                                                 *
*                                                                     *
*                                                                     *
*     ALL PARAMETERS ARE PASSED IN COMMON BLOCKS.                     *
*     VARIABLES USED ARE:                                             *
*                                                                     *
*        ELLIP  :  'YES' OR 'NO ', DEPENDING ON WHETHER THE SHAPE     *
*                  WAS COMPUTED                                       *
*        TYPE   :  CHARACTER STRING 'FLAT    ' OR 'CAMBERED'          *
*        ITYPE  :  INTEGER, 1 IF FLAT, 2 IF CAMBERED                  *
*        TITLE  :  THE TITLE OF THE GRAPH, 6 POSSIBLE TITLES ARE USED *
*        ISELEC :  INTEGER RESPONSE FROM THE GRAPHIC SCREEN           *
*        XMIN   :  MIN VALUE ON HORIZONTAL AXIS FOR PLOT OF A SECTION *
*       XMAX   :  MAX VALUE ON HORIZONTAL AXIS FOR PLOT OF A SECTION *
*       YMIN   :  MIN VALUE ON HORIZONTAL AXIS FOR PLOT OF A SPAN    *
*       YMAX   :  MAX VALUE ON HORIZONTAL AXIS FOR PLOT OF A SPAN    *
*       VMAX   :  MAX VALUE ON VERTICAL AXIS OF PLOT                 *
*       VMIN   :  MIN VALUE ON VERTICAL AXIS OF PLOT                 *
*       VERT   :  A TEMPORARY VARIABLE USED TO HOLD A VALUE.         *
*       IOFFS  :  AN OFFSET USED TO LOCATE THE CORRECT COLUMN IN THE *
*                 WING MATRIX                                        *
*       ISECT  :  SPANWISE SECTION TO BE PLOTTED                     *
*       ESC    :  THE ESCAPE CHARACTER (27)                          *
*       HSTART :  POSITION ON THE GRAPH WHERE THE HORIZONAL AXIS     *
*                 STARTS                                             *
*       VINC   :  VERTICLE AXIS INCREMENT IN UNITS                   *
*       XINC   :  HORIZONTAL AXIS INCREMENT WHEN PLOTTING CHORD      *
*       YINC   :  HORIZONTAL AXIS INCREMENT WHEN PLOTTING SPAN       *
*       PLE    :  PLOT LEADING EDGE POSITION IN INCHES               *
*       PTE    :  PLOT TRAILING EDGE POSITION IN INCHES              *
*       XLE    :  LEADING EDGE POSITION IN PLANFORM UNITS            *
*       XTE    :  TRAILING EDGE POSITION IN PLANFORM UNITS           *
*       AXIS   :  CHARACTER ARRAY WHICH HOLDS THE TITLES FOR THE     *
*                 HORIZONTAL AXIS.                                   *
*       TITLE  :  CHARACTER ARRAY WHICH HOLDS THE TITLES FOR THE     *
*                 VERTICAL AXIS.                                     *
*       TITLE2 :  CHARACTER ARRAY WHICH HOLDS PART OF THE GRAPH      *
*                 TITLE.                                             *
*       TYPE   :  CHARACTER ARRAY WHICH HOLDS PART OF THE GRAPH      *
*                  TITLE                                               *
*        COEF   :  CHARACTER ARRAY WHICH HOLDS PART OF THE GRAPH       *
*                  TITLE.                                              *
*        COEFF  :  REAL ARRAY WHICH HOLDS THE LIFT, DRAG OR MOMENT     *
*                  ASSOCIATED WITH A PARTICULAR GRAPH.                 *
*                                                                      *
*     SUBROUTINES CALLED ARE:                                          *
*        CDAT85 ,  FORTRAN, GENERAL DATA ENTRY                         *
*        DELAY  ,  FORTRAN, SLOW DOWN THE PROGRAM SO ERROR MESSAGES    *
*                   CAN BE SEEN BEFORE BEING ERASED.                    *

*        SDIG   ,  FORTRAN, USED TO FIND THE NUMBER OF SIGNIFICANT     *
*                  DIGITS IN THE AXIS LIMITS AND MAKE AXIS LIMITS      *
*                  MORE UNIFORM.                                       *
*        RNDUP  ,  FORTRAN, USED TO ROUND THE AXIS LIMITS TO THE NEXT  *
*                  WHOLE DIGIT.                                        *
*                                                                      *
*                                                                      *
***********************************************************************
      PROGRAM GRAP85
      REAL LAMDA
      CHARACTER*3 ELLIP
      CHARACTER*8 TYPE(2)
      CHARACTER*20 TITLE(6)
      CHARACTER*8 TITLE2(2,2)
      CHARACTER*1 ESC
      CHARACTER*23 AXIS(2)
      CHARACTER*7 COEF(2,3)
      REAL COEFF(2,4)
      COMMON PI,B,WING(101,23), SECTN(11,10),AR,LAMDA,DELTA,NX,NY,
     * ALPHA,CLDSRD,ELLIP,XAC,CLA
      COMMON/COF/A(101,101), NEQNS
      COMMON/CAM/AA(101,101), AS(101,101)
      COMMON/FILS/IIN,IOUT,JOUT,KOUT,LOUT,MOUT,NOUT
************************************************************************
*                                                                      *
*     DEFINE THE CHARACTER STRINGS NEEDED FOR GRAPH TITLES             *
*                                                                      *
************************************************************************
      TYPE(1)='    FLAT'
      TYPE(2)='CAMBERED'
      TITLE(1)= 'd(CL)/dy            '
      TITLE(2)= 'd(CD)/dy            '
      TITLE(3)= 'd(CMAC)/dy          '
      TITLE(4)= 'TWIST IN DEGREES    '
      TITLE(5)= 'DELTA CP            '
      TITLE(6)= 'WING HEIGHT COORD Z '
      TITLE2(1,1)= ', AOA = '
      TITLE2(1,2)= ' deg    '
      TITLE2(2,1)= ', CLi = '
      TITLE2(2,2)= '        '
      AXIS(1)=  'SPAN-WISE COORDINATE Y '
      AXIS(2)=  'CHORD-WISE COORDINATE X'
      COEF(1,1)='CL   = '
      COEF(1,2)='CD   = '
      COEF(1,3)='CMAC = '
      COEF(2,1)='CLi  = '
      COEF(2,2)='CD   = '
      COEF(2,3)='CMAC = '
************************************************************************
*                                                                      *
*     ASSIGN DEFAULT VALUES TO THE VARIABLES                           *
*                                                                      *
************************************************************************
      IMOD =  1
      AR =    2.0
      LAMDA = 1.0
      ADELTA= 1.0
      NX =    4
      NY =    4
      ALPHA = 5.0
      ELLIP = 'YES'
      CLDSRD= 1.0
      ITYPE = 1
      ISECT = 1
      XAC =   0.0
      ESC = CHAR(27)
      PI = 2.0*ACOS(0.0)
```

```
************************************************************************
*                                                                      *
*     SET INPUT/OUTPUT VARIABLES AND OPEN THE NECESSARY FILES          *
*                                                                      *
************************************************************************
      IIN = 5
      IOUT = 6
      JOUT = 10
      KOUT = 11
      LOUT = 12
      LDAT1 = 13
      LDAT2 = 14
      OPEN(UNIT = IIN)
      OPEN(UNIT = IOUT)
      OPEN(UNIT = JOUT, FILE='WING.MAT')
      OPEN(UNIT = KOUT, FILE='SECTION.MAT')
      OPEN(UNIT = LOUT, FILE='PARAMS.DAT')
      OPEN(UNIT = LDAT1, FILE='FLAT.DAT')
      OPEN(UNIT = LDAT2, FILE='CAMBER.DAT')
10    FORMAT (I6/3(F6.2,/),2(I3/),F6.2/A/F6.2/I2/I2/F6.2)
20    FORMAT (/2(10X,F12.6/)/10X,F12.6)
************************************************************************
*                                                                      *
*     FORMAT STATEMENTS ARE FOR THE INPUT TO THE GRAPHER PROGRAM.      *
*     OTHER GRAPHICS PROGRAMS WILL REQUIRE DIFFERENT FORMAT STMTS.     *
*                                                                      *
```

```
********************************************************************************

32     FORMAT (

     * '1236'/

     * '1 2 1 0 0'/

     * 'P1'/

     * '65 66 78 "NO " 0'/

     * '"NO" "SOLID" 1.500e-001 1'/

     * '"YES" 41 1.000e-001 1 1'/

     * '78 9.900e+028 9.900e+028 0.000e+000 "DEFAULT" 1.000e-001 1'/

     * '"SOLID" 5 1.500e-001 0.000e+000 9.900e+029 100 2.000e+000 1')

33     FORMAT (

     * '1236'/

     * '1 2 3 0 1'/

     * 'P1'/

     * '65 66 78 "NO " 0'/

     * '"NO" "SOLID" 1.500e-001 1'/

     * '"YES" 41 1.000e-001 1 1'/

     * '78 9.900e+028 9.900e+028 0.000e+000 "DEFAULT" 1.000e-001 1'/

     * '"SOLID" 5 1.500e-001 0.000e+000 9.900e+029 100 2.000e+000 1')

34     FORMAT(

     * 'HORIZ'/

     * '1.750e+000 ',E10.3,' 6.000e+000 88'/

     * E10.3,' ',E10.3,' 9.900e+028 1 1'/

     * '0.000e+000 ',E10.3,' 1.500e-001 1 1'/

     * '1 1 1'/

     * '1 2.750e+000 4.000e-001 9.900e+028 1.800e-001'/
     * '"DEFAULT" "DEFAULT" "',A,'"')
36    FORMAT(
     * 'Y-AXIS'/
     * '1.750e+000 1.000e+000 5.000e+000 89'/
     * E10.3,' ',E10.3,' 9.900e+028 1 1'/
     * '2.700e+002 ',E10.3,' 1.500e-001 1 1'/
     * '1 ',I1,' 1'/
     * '1 9.900e+028 0.000e+000 9.900e+028 1.800e-001'/
     * '"DEFAULT" "DEFAULT" "',A,'"')
37    FORMAT(
     * 'P1'/
     * '"DEFAULT" 1'/
     * '2.000e+000 7.000e+000 0.000e+000 1.700e-001'/
     * '2'/
     * '0 13 "',A,' WING"'/
     * '1 12 "CLi  = ',F5.2,'"')
38    FORMAT(
     * 'P1'/
     * '"DEFAULT" 1'/
     * '2.000e+000 7.000e+000 0.000e+000 1.700e-001'/
     * '2'/
     * '0 13 "',A,' WING"'/
     * '1 34 "',A,F6.4,A,F5.2,A'"')
39    FORMAT(
     * 'P1'/
     * '"DEFAULT" 1'/
     * '2.000e+000 7.000e+000 0.000e+000 1.700e-001'/
     * '2'/
     * '0 26 "',A,' WING, SECTION ',I3,'"'/
     * '1 26 "vertical scale exaggerated"')
40    FORMAT(
     * 'P1'/
     * '"DEFAULT" 1'/
     * '2.000e+000 7.000e+000 0.000e+000 1.700e-001'/
     * '1'/
     * '0 26 "',A,' WING, SECTION ',I3,'"')
41    FORMAT(
     * 'LE'/
     * '"DEFAULT" 1'/
     * E10.3,' 6.000e+000 0.000e+000 1.000e-001'/
     * '1'/
     * '0 4 "L.E."')
42    FORMAT(
     * 'TE'/
     * '"DEFAULT" 1'/
     * E10.3,' 6.000e+000 0.000e+000 1.000e-001'/
     * '1'/
     * '0 4 "T.E."')
43    FORMAT(
     * 'P1'/
     * '2'/
     * E10.3,' 1.000e+000 ',E10.3,' 6.000e+000 1 2 1.000e-001'/
     * E10.3,' 1.000e+000 ',E10.3,' 6.000e+000 1 2 1.000e-001')
50     FORMAT (2(F10.6,1X))
60     FORMAT (23F10.6)
70     FORMAT (10F10.6)
***********************************************************************
*                                                                     *
*      THESE FORMAT STATEMENTS ARE FOR THE FLASHUP WINDOWS MENUS      *
*                                                                     *
***********************************************************************
82     FORMAT (' ',A,A,I3)
84     FORMAT (' ',A,A,A)
***********************************************************************
*                                                                     *
*      READ THE LAST SET OF PARAMETERS, IF UNCHANGED, READ WING AND   *
*      SECTN DATA , IMOD IS THE FLAG TO TELL IF THE PARAMETERS HAVE   *
*      BEEN CHANGED.  1 = CHANGE, 0 = NO CHANGE.                      *
*                                                                     *
***********************************************************************
       READ (LOUT,10) IMOD,AR,LAMDA,ADELTA,NX,NY,ALPHA,
     * ELLIP,CLDSRD,ITYPE,ISECT,XAC
       DELTA = ADELTA*PI/180.
       NEQNS = NX*NY
       B = AR*(1.0+LAMDA)/2.0
       IF (IMOD.EQ.0) THEN
          READ (JOUT,60) ((WING(I,J),J=1,23),I=1,NEQNS)
          READ (KOUT,70) ((SECTN(I,J),J=1,10),I=1,NY)
```

```
         READ (LDAT1,20) (COEFF(1,J),J=1,3)
         READ (LDAT2,20) (COEFF(2,J),J=1,3)
      ELSE
         WRITE (IOUT,*) ' CHANGES WERE MADE TO THE PLANFORM. GO BACK'
         WRITE (IOUT,*) ' AND RUN THE LOAD/SHAPE PROGRAM AGAIN.'
         CALL DELAY (6)
         GO TO 1000
      ENDIF
      COEFF(1,4) = ALPHA
      COEFF(2,4) = CLDSRD
      CLOSE (JOUT)
      CLOSE (KOUT)
      CLOSE (LDAT1)
      CLOSE (LDAT2)
************************************************************************
*                                                                      *
*     SET MINIMUM VALUE OF X.  THIS X IS THE CHORDWISE COORDINATE      *
*                                                                      *
************************************************************************
      XMIN = 0.0
************************************************************************
*                                                                      *
*     COMPUTE THE CORNER  OF THE WING, TO GET MAX VALUES FOR THE AXIS  *
*     THIS IS NEEDED IN CASE THE WING IS SWEPT OR HIGHLY TAPERED.      *
*                                                                      *
************************************************************************
```

```
      XT = (B*TAN(DELTA)/2.0) + LAMDA
***********************************************************************
*                                                                     *
*     NOW CONVERT SPAN AND CHORD AXIS LIMITS INTO "ROUNDED" UNITS     *
*     AT THE SAME TIME, COMPUTE THE INCREMENTS IN TIC MARKS FOR THE   *
*     GRAPHS.                                                         *
*                                                                     *
***********************************************************************

      IF (XT.GT.1.) THEN
         CALL RNDUP(XT,XMAX)
      ELSE
         XMAX = 1.0
      ENDIF
      YMIN = 0.0
      CALL RNDUP(B/2.0,YMAX)
      XINC = (XMAX-XMIN)/5.0
      YINC = (YMAX-YMIN)/5.0
***********************************************************************
*                                                                     *
*     CLEAR THE SCREEN AND WRITE THE GRAPHIC MENU.                    *
*                                                                     *
***********************************************************************
110   WRITE (IOUT,*) ESC,'[2J'
      WRITE (IOUT,*) ESC,'[00;00H'
      WRITE (IOUT,*) '~C=ALL/'
      WRITE (IOUT,*) '~W=GRAPH/'
      WRITE (IOUT,84) ESC,'[08;69H',TYPE(ITYPE)
      WRITE (IOUT,82) ESC,'[09;69H',ISECT
***********************************************************************
*                                                                     *
*     READ THE OPTION AND SEND TO THE APPROPRIATE AREA.               *
*     OPTIONS ARE:                                                    *
*        1, SETS THE TYPE OF WING, FLAT OR ELLIPTIC.                  *
*        2, SETS THE WING SECTION TO BE PLOTTED                       *
*        3, RETURNS TO THE MAIN MENU                                  *
*             4 - 7 PLOT THE LIFT, DRAG, MOMENT  OR TWIST VS SPAN     *
*             8     PLOT DELTA CP VERSUS CHORD                        *
*             9     PLOTS THE WING SHAPE VERSUS CHORD, WHICH GIVES    *
*                   AN ELLIPTIC LIFT DISTRIBUTION.                    *
*                                                                     *
***********************************************************************
      READ (IIN,*) ISELEC
      WRITE (IOUT,*) ESC,'[2J'
      WRITE (IOUT,*) ESC,'[00;00H'
***********************************************************************
*                                                                     *
*     THIS IS OPTION 1, SET TYPE OF WING.                             *
*                                                                     *
***********************************************************************
      IF (ISELEC.EQ.1) THEN
         IF (ELLIP.EQ.'NO ') THEN
```

```
              WRITE (IOUT,*) ' ELLIPTIC DATA WAS NOT COMPUTED'
              CALL DELAY(6)
              GO TO 110
         ELSE
         ENDIF
         IF (ITYPE.EQ.1) THEN
              ITYPE = 2
         ELSE
              ITYPE = 1
         ENDIF
         GO TO 110
      ELSE
      ENDIF
************************************************************************
*                                                                      *
*     THIS IS OPTION 2, SET SECTION NUMBER                             *
*                                                                      *
************************************************************************
      IF (ISELEC.EQ.2) THEN
         CALL CDIT85(1,NY,ISECT,
     *  'WING SECTION NUMBER                 ',IMOD)
         IMOD = 0
         GO TO 110
      ELSE
      ENDIF
************************************************************************
```

```
*                                                                      *
*     THIS IS OPTION 3, RETURN TO MAIN PROGRAM. BEFORE LEAVING, WRITE *
*     THE FLASHUP WINDOW FOR THE MAIN MENU.                            *
*                                                                      *
************************************************************************
      IF (ISELEC.EQ.3) THEN
         GO TO 1000
      ELSE
      ENDIF
      IF ((ISELEC.LT.1).OR.(ISELEC.GT.9)) THEN
         WRITE (IOUT,*) ' THAT IS NOT A VALID SELECTION'
         CALL DELAY(6)
         GO TO 110
      ELSE
      ENDIF
************************************************************************
*                                                                      *
*     THESE ARE THE LIFT, DRAG , MOMENT AND TWIST VS SPAN PLOTS.       *
*                                                                      *
************************************************************************
      IF (ISELEC.GT.7) GO TO 310
************************************************************************
*                                                                      *
*     OFFSET IS ESTABLISHED, IT IS AN EASY WAY TO USE THE ISELEC       *
*     TO FIND THE CORRECT COLUMN OF THE SECTN MATRIX.                  *
*     SINCE THERE IS NO TWIST ON A FLAT WING, SET THE WING TYPE TO     *
*     CAMBERED IF THE TWIST PLOT IS SELECTED.                          *
*                                                                      *
************************************************************************
      IF (ISELEC.EQ.7) ITYPE = 2
      IF (ITYPE.EQ.1) THEN
         IOFFS = -2
      ELSE
         IOFFS = 3
      ENDIF
************************************************************************
*                                                                      *
*     SET UP THE DATA FILE FOR THE GRAPHIC PLOT.                       *
*                                                                      *
************************************************************************
      J = 1
      VMAX = -100.0
      VMIN = 100.0
      OPEN(UNIT = 30,FILE ='P1.DAT')
      IJ = J
      DO 300 I = 1,NY
         IJ = (I-1)*NX + J
         VERT = SECTN(I,IOFFS+ISELEC)
         IF (VMAX.LT.VERT) VMAX = VERT
         IF (VMIN.GT.VERT) VMIN = VERT
         WRITE (30,50) WING(IJ,5), VERT
300   CONTINUE
```

```
      CLOSE (30)
************************************************************************
*                                                                      *
*     SET UP THE P1.GRF FILE FOR GRAPHER.  IT HAS ALL THE GRAPH        *
*     FORMATS IN IT.                                                   *
*                                                                      *
************************************************************************
      IF (VMIN.GT.0.0) VMIN = 0.0
      IF (VMAX.LT.0.0) VMAX = 0.0
      CALL RNDUP(VMAX,VMAX)
      CALL RNDUP(VMIN,VMIN)
      CALL SDIG(VMIN,VMIN,VMAX,VMAX,IDIG)
      VINC = (VMAX-VMIN)/5.0
      HSTART =(VMAX-6*VMIN)/(VMAX-VMIN)
      OPEN(UNIT = 32,FILE ='P1.GRF')
      WRITE (32,32)
      WRITE (32,34) HSTART,YMIN,YMAX,YINC,AXIS(1)
      WRITE (32,36) VMIN,VMAX,VINC,IDIG,TITLE(ISELEC-3)
      IF (ISELEC.EQ.7) THEN
         WRITE (32,37) TYPE(ITYPE),COEFF(2,4)
      ELSE
         WRITE (32,38) TYPE(ITYPE),COEF(ITYPE,ISELEC-3),
     *   COEFF(ITYPE,ISELEC-3), TITLE2(ITYPE,1), COEFF(ITYPE,4),
     *   TITLE2(ITYPE,2)
      ENDIF
      CLOSE (32)
      GO TO 1000
************************************************************************
*                                                                      *
*     THESE ARE THE DELTA CP PLOTS                                     *
*                                                                      *
************************************************************************
310   CONTINUE
      IF (ISELEC.EQ.9) GO TO 400
************************************************************************
*                                                                      *
*     OFFSET IS SET AGAIN.                                             *
*                                                                      *
************************************************************************
      IF (ITYPE.EQ.1) THEN
         IOFFS = 8
      ELSE
         IOFFS = 10
      ENDIF
************************************************************************
*                                                                      *
*     SET UP THE P1.DAT FILE FOR GRAPHER                               *
*                                                                      *
************************************************************************
      I = ISECT
      VMAX = -100.0
      VMIN = 100.0
      OPEN(UNIT = 30,FILE ='P1.DAT')
      IJ = (I-1)*NX + 1
************************************************************************
*                                                                      *
*     XLE IS THE VALUE OF THE LEADING EDGE.  SINCE THE DELTA CP        *
*     FOR THE CAMBERED WING IS ZERO AT THE LEADING AND TRAILING        *
*     EDGE, IT IS POSSIBLE TO EXTEND THE DATA TO THOSE POINTS.         *
*                                                                      *
************************************************************************
      XLE = WING(IJ,4)-WING(IJ,13)/2.
      IF(ITYPE.EQ.2) THEN
         WRITE (30,50) XLE, 0.0
      ELSE
      ENDIF
************************************************************************
*                                                                      *
*     GET THE MAX AND MIN VALUES OF THE PARAMETER BEING PLOTTED        *
*     SO ALL GRAPHS WITH THAT PARAMETER WILL HAVE THE SAME VERTICAL    *
*     SCALE.                                                           *
*                                                                      *
************************************************************************
      DO 315 IJ = 1,NEQNS
         VERT = WING(IJ,IOFFS+ISELEC)
         IF (VMAX.LT.VERT) VMAX = VERT
         IF (VMIN.GT.VERT) VMIN = VERT
315   CONTINUE
```

```
**********************************************************************
*                                                                    *
*     SET UP THE P1.DAT FILE FOR GRAPHER                             *
*                                                                    *
**********************************************************************
      DO 320 J = 1,NX
         IJ = (I-1)*NX +J
         VERT = WING(IJ,IOFFS+ISELEC)
         WRITE (30.50) WING(IJ,11), VERT
320   CONTINUE
      IJ = I*NX
**********************************************************************
*                                                                    *
*     XLE IS COMPUTED FOR THE REASONS STATED ABOVE.                  *
*                                                                    *
**********************************************************************
      XTE = WING(IJ,4)+WING(IJ,13)/2.
      IF(ITYPE.EQ.2) THEN
         WRITE (30,50) XTE, 0.0
      ELSE
      ENDIF
      CLOSE (30)
**********************************************************************
*                                                                    *
*     COMPUTE AND STORE THE VALUES FOR THE P1.GRF FILE USED BY GRAPHER*
*                                                                    *
```

```
************************************************************************
      IF (VMIN.GT.0.0) VMIN = 0.0
      IF (VMAX.LT.0.0) VMAX = 0.0
      CALL RNDUP(VMAX,VMAX)
      CALL RNDUP(VMIN,VMIN)
      CALL SDIG(VMIN,VMIN,VMAX,VMAX,IDIG)
      VINC = (VMAX-VMIN)/5.0
      HSTART = (VMAX-6*VMIN)/(VMAX-VMIN)
      PLE = (6.*XLE/(XMAX-XMIN))+1.75
      PTE = (6.*XTE/(XMAX-XMIN))+1.75
      OPEN(UNIT = 32,FILE ='P1.GRF')
      WRITE (32,33)
      WRITE (32,34) HSTART,XMIN,XMAX,XINC,AXIS(2)
      WRITE (32,36) VMIN,VMAX,VINC,IDIG,TITLE(ISELEC-3)
      WRITE (32,40) TYPE(ITYPE),ISECT
      WRITE (32,41) PLE
      WRITE (32,42) PTE
      WRITE (32,43) PLE,PLE,PTE,PTE
      CLOSE (32)
      GO TO 1000
************************************************************************
*                                                                      *
*     THIS IS THE SHAPE PLOT.                                          *
*                                                                      *
************************************************************************
400   IF (ELLIP.EQ.'NO ') THEN
```

```
           WRITE (IOUT,*) ' ELLIPTIC SHAPE DATA WAS NOT COMPUTED'
           CALL DELAY(6)
           GO TO 110
      ELSE
      ENDIF
************************************************************************
*                                                                      *
*     SET UP THE P1.DAT FILE FOR THE WING SHAPE.  WE KNOW THAT THE     *
*     LEADING EDGE STARTS AT ZERO HEIGHT, SO THAT STARTING VALUE IS    *
*     ADDED TO THE DATA, IN ORDER TO GET THE MAXIMUM INFORMATION       *
*     INCLUDED IN THE GRAPH                                            *
*     XLE AND XTE ARE ALSO NEEDED SO THAT VERTICAL DOTTED LINES        *
*     SHOWING THEIR POSITION CAN BE ADDED TO THE GRAPH.                *
*                                                                      *
************************************************************************
      I = ISECT
      VMAX = -100.0
      VMIN = 100.0
      OPEN(UNIT = 30,FILE ='P1.DAT')
      IJ = (I-1)*NX + 1
      XLE = WING(IJ,4)-WING(IJ,13)/2.
      WRITE (30,50) XLE, 0.0
      IJ = I*NX
      XTE = WING(IJ,4)+WING(IJ,13)/2.
      DO 405 IJ = 1,NEQNS
         VERT = WING(IJ,20)
```

```
               IF (VMAX.LT.VERT) VMAX = VERT
               IF (VMIN.GT.VERT) VMIN = VERT
 405       CONTINUE
           DO 410 J = 1,NX
               IJ = (I-1)*NX+J
               VERT = WING(IJ,20)
               WRITE (30,50) WING(IJ,4)+WING(IJ,13)/2.0, VERT
 410       CONTINUE
           CLOSE (30)
           IF (VMIN.GT.0.0) VMIN = 0.0
           IF (VMAX.LT.0.0) VMAX = 0.0
           CALL RNDUP(VMAX,VMAX)
           CALL RNDUP(VMIN,VMIN)
           CALL SDIG(VMIN,VMIN,VMAX,VMAX,IDIG)
           VINC = (VMAX-VMIN)/5.0
           HSTART = (VMAX-6*VMIN)/(VMAX-VMIN)
           PLE = (6.*XLE/(XMAX-XMIN))+1.75
           PTE = (6.*XTE/(XMAX-XMIN))+1.75
           OPEN(UNIT = 32,FILE ='P1.GRF')
           WRITE (32,33)
           WRITE (32,34) HSTART,XMIN,XMAX,XINC,AXIS(2)
           WRITE (32,36) VMIN,VMAX,VINC,IDIG,TITLE(ISELEC-3)
           WRITE (32,39) TYPE(2),ISECT
           WRITE (32,41) PLE
           WRITE (32,42) PTE
           WRITE (32,43) PLE,PLE,PTE,PTE
```

```
      CLOSE (32)
1000  REWIND (LOUT)
*************************************************************************
*                                                                       *
*     RE-WRITE THE PARAMETER DATA FILE, SINCE IT KEEPS TRACK OF THE     *
*     LAST WING TYPE AND SECTION PLOTTED.                               *
*                                                                       *
*************************************************************************
      WRITE (LOUT,10) IMOD,AR,LAMDA,ADELTA,NX,NY,ALPHA,
     *ELLIP,CLDSRD,ITYPE,ISECT,XAC
      CLOSE (LOUT)
*************************************************************************
*                                                                       *
*     BEFORE LEAVING THE PROGRAM, RE-WRITE THE MAIN PROGRAM MENU.       *
*                                                                       *
*************************************************************************
      WRITE (IOUT,*) ESC,'[2J'
      WRITE (IOUT,*) ESC,'[00;00H'
      WRITE (IOUT,*) '~C=ALL/'
      WRITE (IOUT,*) '~W=MAIN/'
      END
$INCLUDE:'CDIT85.FOR'
$INCLUDE:'DELAY.FOR'
$INCLUDE:'RNDUP.FOR'
$INCLUDE:'SDIG.FOR'
```

DELAY

```
************************************************************************
*                                                                      *
*     SUBROUTINE DELAY(INTERV)                                         *
*                                                                      *
*     CALLED TO SLOW THE SCREEN DOWN AFTER PRINTING AN ERROR MESSAGE.  *
*     VARIABLES USED ARE:                                              *
*        INTERV :  NUMBER OF SECONDS FOR DELAY MUST BE LESS THAN 59,   *
*                  INPUT.                                              *
*        IHR    :  TIME OF DAY IN HOURS                                *
*        IMIN   :  TIME OF DAY IN MIN                                  *
*        ISEC1  :  TIME OF DAY IN SECONDS                              *
*        ISEC2  :  TIME OF DAY IN SECONDS                              *
*        IHUND  :  TIME OF DAY IN HUNDREDTHS OF SECONDS.               *
*                                                                      *
************************************************************************
      SUBROUTINE DELAY(INTERV)
      INTEGER*2 IHR, IMIN, ISEC1, ISEC2, IHUND, INTER
      INTER = INTERV
      CALL GETTIM(IHR,IMIN,ISEC1,IHUND)
100   CALL GETTIM(IHR,IMIN,ISEC2,IHUND)
      IF ((ISEC2-ISEC1).LT.INTER) GOTO 100
      RETURN
      END
```

CDIT85

```
******************************************************************
*                                                                *
*     SUBROUTINE CDIT85(LOW,HIGH,VALUE,NAME,IMOD)                *
*     INTEGER VERSION OF CDAT85                                  *
*     HANDLE INPUT OF PARAMETER DATA FOR THE PROGRAM.  INCLUDES SOME *
*     ERROR CHECKING.                                            *
*                                                                *
*     VARIABLES USED ARE:                                        *
*        LOW     :  MINIMUM VALUE THAT WILL BE ACCEPTED, INPUT   *
*        HIGH    :  MAXIMUM VALUE THAT WILL BE ACCEPTED, INPUT   *
*        VALUE   :  VALUE READ FROM SCREEN, OUTPUT               *
*        NAME    :  SCREEN PROMPT STRING, INPUT                  *
*        IMOD    :  FLAG USED IN ANOTHER PROGRAM, SET TO 1 ANYTIME  *
*                     CDAT85 IS CALLED., OUTPUT                    *

*        ESC     :  ESCAPE CHARACTER, (27)                       *
*                                                                *
******************************************************************
      SUBROUTINE CDIT85(LOW,HIGH,VALUE,NAME,IMOD)
      COMMON/FILS/IIN,IOUT,JOUT,KOUT,LOUT,MOUT,NOUT
      INTEGER LOW,HIGH,VALUE
      CHARACTER*35 NAME
      CHARACTER*1 ESC
      ESC = CHAR(27)
10    FORMAT(' ENTER THE ',A35)
```

```
20    FORMAT(' THE MINIMUM VALUE FOR ',A35,/' IS ',I3)
30    FORMAT(' THE MAXIMUM VALUE FOR ',A35,/' IS ',I3)
************************************************************************
*                                                                      *
*     CLEAR SCREEN, POSITION CURSOR, AND                               *
*     SET IMOD FLAG AND WRITE PROMPT TO SCREEN.                        *
*                                                                      *
************************************************************************
100   WRITE (IOUT,*) ESC,'[2J'
      WRITE (IOUT,*) ESC,'[10;0H'
      IMOD = 1
      WRITE(IOUT,10) NAME
************************************************************************
*                                                                      *
*     READ SCREEN VALUE.  ON ERROR OR END OF DATA SET, GO BACK TO      *
*     PROMPT.                                                          *
*                                                                      *
************************************************************************
      READ(IIN,*,ERR = 100, END = 100) VALUE
************************************************************************
*                                                                      *
*     IF VALUES ARE OUT OF RANGE , PRINT ERROR MSG AND RETURN TO       *
*     PROMPT.                                                          *
*                                                                      *
************************************************************************
      IF (VALUE.LT.LOW) THEN
```

```
   WRITE (IOUT,20) NAME,LOW
   CALL DELAY (6)
   GO TO 100
ELSE
   IF(VALUE.GT.HIGH) THEN
        WRITE (IOUT,30) NAME,HIGH
        CALL DELAY(6)
        GO TO 100
     ELSE
     END IF
END IF
RETURN
END
```

CDAT85

```
**********************************************************************
*                                                                    *
*     SUBROUTINE CDAT85(LOW,HIGH,VALUE,NAME,IMOD)                    *
*     REAL VERSION                                                   *
*     HANDLE INPUT OF PARAMETER DATA FOR THE PROGRAM.  INCLUDES SOME *
*     ERROR CHECKING.                                                *
*                                                                    *
*     VARIABLES USED ARE:                                            *
*        LOW    :  MINIMUM VALUE THAT WILL BE ACCEPTED, INPUT        *
*        HIGH   :  MAXIMUM VALUE THAT WILL BE ACCEPTED, INPUT        *
*        VALUE  :  VALUE READ FROM SCREEN, OUTPUT                    *
*        NAME   :  SCREEN PROMPT STRING, INPUT                       *
*        IMOD   :  FLAG USED IN ANOTHER PROGRAM, SET TO 1 ANYTIME    *
*                  CDAT85 IS CALLED., OUTPUT                         *
*        ESC    :  ESCAPE CHARACTER, (27)                            *
*                                                                    *
**********************************************************************
      SUBROUTINE CDAT85(LOW,HIGH,VALUE,NAME,IMOD)
      COMMON/FILS/IIN,IOUT,JOUT,KOUT,LOUT,MOUT,NOUT
      REAL LOW
      CHARACTER*35 NAME
      CHARACTER*1 ESC
      ESC = CHAR(27)
10    FORMAT(' ENTER THE ',A35)
20    FORMAT(' THE MINIMUM VALUE FOR ',A35,/' IS ',F6.2)
```

```
30    FORMAT(' THE MAXIMUM VALUE FOR ',A35,/' IS ',F6.2)
***********************************************************************
*                                                                     *
*     CLEAR SCREEN, POSITION CURSOR, AND                              *
*     SET IMOD FLAG AND WRITE PROMPT TO SCREEN.                       *
*                                                                     *
***********************************************************************
100   WRITE (IOUT,*) ESC,'[2J'
      WRITE (IOUT,*) ESC,'[10;0H'
      IMOD = 1
      WRITE(IOUT,10) NAME
***********************************************************************
*                                                                     *
*     READ SCREEN VALUE.  ON ERROR OR END OF DATA SET, GO BACK TO     *
*     PROMPT.                                                         *
*                                                                     *
***********************************************************************
      READ(IIN,*,ERR = 100, END = 100) VALUE
***********************************************************************
*                                                                     *
*     IF VALUES ARE OUT OF RANGE , PRINT ERROR MSG AND RETURN TO      *
*     PROMPT.                                                         *
*                                                                     *
***********************************************************************
      IF (VALUE.LT.LOW) THEN
         WRITE (IOUT,20) NAME,LOW
```

```
   CALL DELAY (6)
   GO TO 100
ELSE
   IF(VALUE.GT.HIGH) THEN
        WRITE (IOUT,30) NAME,HIGH
        CALL DELAY (6)
        GO TO 100
     ELSE
     END IF
END IF
RETURN
END
```

GAUSS

```
************************************************************************
*                                                                      *
*     SUBROUTINE GAUSS(NRHS)                                           *
*                                                                      *
*     SOLUTION OF LINEAR ALGEGRAIC SYSTEM BY GAUSS ELIMINATION          *
*     WITH PARTIAL PIVOTING.  TAKEN FROM 'AN INTRODUCTION TO           *
*     THEORETICAL AND COMPUTATIONAL AERODYNAMICS', JACK MORAN,         *
*     JOHN WILEY & SONS, NEW YORK, 1984, PG 78.                        *
*                                                                      *
*     VARIABLES USED ARE:                                              *
*        A()     :  COEFFICIENT MATRIX, AUGMENTED WITH RIGHT HAND      *
*                   SIDES IN COLUMN NEQNS+1, INPUT/OUTPUT, UNUSABLE    *
*                   AFTER RETURN                                       *
*        NEQNS   :  NUMBER OF EQUATIONS IN MATRIX, INPUT               *
*        NRHS    :  NUMBER OF RIGHT HAND SIDES, INPUT                  *
*                                                                      *
************************************************************************
      SUBROUTINE GAUSS(NRHS)
      COMMON/COF/A(101,101),NEQNS
      NP = NEQNS+1
      NTOT = NEQNS+NRHS
************************************************************************
*                                                                      *
*     SEARCH FOR THE LARGEST ENTRY IN THE (I-1)TH COLUMN, ON OR        *
*     BELOW THE MAIN DIAGONAL.                                         *
*                                                                    *
**********************************************************************
      DO 150 I = 2,NEQNS
         IM = I-1
         IMAX = IM
         AMAX = ABS(A(IM,IM))
         DO 110 J = I,NEQNS
              IF (AMAX.GE.ABS(A(J,IM))) GO TO 110
              IMAX = J
              AMAX = ABS(A(J,IM))
110      CONTINUE
**********************************************************************
*                                                                    *
*     SWITCH THE (I-1)TH AND IMAXTH EQUATIONS                        *
*                                                                    *
**********************************************************************
         IF (IMAX.NE.IM) GO TO 140
         DO 130 J = IM,NTOT
              TEMP = A(IM,J)
              A(IM,J) = A(IMAX,J)
              A(IMAX,J) = TEMP
130      CONTINUE
**********************************************************************
*                                                                    *
*     ELIMINATE THE (I-1)TH UNKNOWN FROM ITH THROUGH (NEQNS)TH       *
*     EQUATIONS.                                                     *
*                                                                              *
********************************************************************************
140   DC 150 J = I,NEQNS
         R = A(J,IM)/A(IM,IM)
         DO 150 K = I,NTOT
         A(J,K) = A(J,K)-R*A(IM,K)
150 CONTINUE
********************************************************************************
*                                                                              *
*   BACK SUBSTITUTE                                                            *
*                                                                              *
********************************************************************************
    DO 220 K = NP,NTOT
      A(NEQNS,K) = A(NEQNS,K)/A(NEQNS,NEQNS)
      DO 210 L = 2,NEQNS
         I = NEQNS+1-L
         IP = I+1
         DO 200 J = IP,NEQNS
            A(I,K) = A(I,K) - A(I,J)*A(J,K)
200      CONTINUE
      A(I,K) = A(I,K)/A(I,I)
210   CONTINUE
220 CONTINUE
    RETURN
    END
```

RNDUP

```
************************************************************************
*                                                                      *
*     SUBROUTINE TO ROUND UP THE VALUE OF AN AXIS LIMIT.               *
*     FOR EXAMPLE IF THE VALUE PASSED IS .0018, THE VALUE RETURNED     *
*     IS .0020.  THIS IS NECESSARY TO ENSURE THAT AXIS LIMITS ARE      *
*     WELL DEFINED.                                                    *
*                                                                      *
*     VARIABLES USED ARE:                                              *
*     VALIN   THE VALUE PASSED                                         *
*     VALOUT  THE VALUE RETURNED                                       *
*     VAL     A TEMPORARY VARIABLE                                     *
*     IRTN    AN INTEGER MULTIPLIER USED TO RETURN TO THE CORRECT      *
*             ORDER OF MAGNITUDE.                                      *
*     ISIDE   AN INTEGER ADDED TO ROUND UP.  DEPENDS ON WHETHER THE    *
*             VALUE PASSED IS POSITIVE OR NEGATIVE.                    *
*     IV      A TEMPORARY VARIABLE                                     *
*                                                                      *
************************************************************************
      SUBROUTINE RNDUP(VALIN,VALOUT)
      RTN = 1.0
      VAL = VALIN
      IF (VAL.EQ.0.0) THEN
         VALOUT = VAL
         GOTO 200
      ELSE
      ENDIF
      IF (VAL.LT.0.0) THEN
         ISIDE = -1
      ELSE
         ISIDE = 1
      ENDIF
100   IF (ABS(VAL).GE.10.0) THEN
         VAL = VAL/10.0
         RTN = RTN*10.0
      ELSE
         IF(ABS(VAL).LT.1.0) THEN
              VAL = VAL*10.0
              RTN = RTN/10.0
         ELSE
              IV = AINT(VAL)+ISIDE
              VALOUT = REAL(IV*RTN)
              GO TO 200
         ENDIF
      ENDIF
      GO TO 100
200   RETURN
      END
```

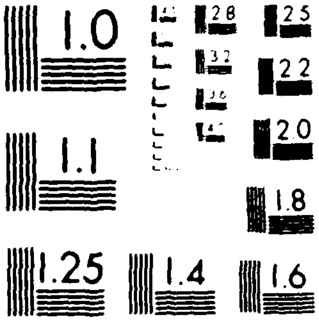
SDIG

```
***********************************************************************
*                                                                     *
*     SUBROUTINE SDIG(MININ,MINOUT,MAXIN,MAXOUT,IDIG)                 *
*                                                                     *
*     SUBROUTINE TO GET THE RANGE OF VALUES IN THE AXIS LIMITS        *
*     TO BE 5 UNITS APART. AND RETURN THE NUMBER OF SIGNIFICANT       *
*     DIGITS FOR THE AXIS.                                            *
*                                                                     *
*     VARIABLES USED ARE:                                             *
*     MININ   THE MINIMUM AXIS VALUE PASSED                           *
*     MINOUT  THE MINIMUM AXIS VALUE RETURNED                         *
*     MAXIN   THE MAXIMUM AXIS VALUE PASSED                           *
*     MAXOUT  THE MAXIMUM AXIS VALUE RETURNED                         *
*     VAL     A TEMPORARY VARIABLE                                    *
*     VALNOT  A TEMPORARY VARIABLE                                    *
*     RTN     A MULTIPLIER USED TO RETURN TO THE CORRECT              *
*             ORDER OF MAGNITUDE.                                     *
*     ISIDE   AN INTEGER TO DETERMINE WHETHER THE VALUE IS POSITIVE   *
*             OR NEGATIVE                                             *
*     IV      A TEMPORARY VARIABLE                                    *
*                                                                     *
***********************************************************************
      SUBROUTINE SDIG(MININ,MINOUT,MAXIN,MAXOUT,IDIG)
      REAL MININ,MINOUT,MAXIN,MAXOUT,RTN,VAL,VALNOT
      INTEGER IDIG,ICASE,ISIDE,IV,ITEST
```

AD-A200 436 AERODYNAMIC PERFORMANCE OF WINGS OF ARBITRARY PLANFORM 3/3
IN INVISCID INCOMPRESSIBLE IRROTATIONAL FLOW(U) NAVAL
POSTGRADUATE SCHOOL MONTEREY CA C L HOLM SEP 88

UNCLASSIFIED F/G 1/1 NL

UNCLA



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

```
      RTN = 1.0
      IF(ABS(MININ).LT.ABS(MAXIN)) THEN
************************************************************************
*                                                                      *
*     ICASE = 1 IS FOR THE MIN AXIS VALUE BEING SMALLER IN MAGNITUDE   *
*     THAN THE MAX AXIS VALUE.                                         *
*                                                                      *
************************************************************************
         ICASE = 1
         VAL = MAXIN
         VALNOT = MININ
      ELSE
************************************************************************
*                                                                      *
*     ICASE = 2 IS FOR THE MAX AXIS VALUE BEING SMALLER IN MAGNITUDE   *
*     THAN THE MIN AXIS VALUE.                                         *
*                                                                      *
************************************************************************
         ICASE = 2
         VAL = MININ
         VALNOT = MAXIN
      ENDIF
************************************************************************
*                                                                      *
*     DETERMINE WHETHER THE SMALLER (IN MAGNITUDE) AXIS VALUE IS       *
*     POSITIVE OR NEGATIVE.                                            *
*                                                                      *
************************************************************************
      ISIDE = SIGN(1,VALNOT)
100   IF (ABS(VAL).GE.10.0) THEN
         VAL = VAL/10.0
         RTN = RTN*10.0
      ELSE
         IF(ABS(VAL).LT.1.0) THEN
              VAL = VAL*10.0
              RTN = RTN/10.0
         ELSE
              IF (VALNOT.EQ.0.0) GOTO 300
              IV = NINT(VALNOT/RTN)
              IF(IV.EQ.0) IV = IV + ISIDE
              GO TO 200
         ENDIF
      ENDIF
      GO TO 100
200   CONTINUE
      ITEST = ABS(NINT(VAL)-IV)
      IF ((ITEST.EQ.5).OR.(ITEST.EQ.10).OR.(ITEST.EQ.15).OR.
     *    (ITEST.EQ.20).OR.(NINT(VAL).EQ.-IV)) GOTO 300
      IV = IV + ISIDE
      ITEST = ABS(NINT(VAL)-IV)
      IF ((ITEST.EQ.5).OR.(ITEST.EQ.10).OR.(ITEST.EQ.15).OR.
     *    (ITEST.EQ.20).OR.(NINT(VAL).EQ.-IV)) GOTO 300
```

```
      VAL = VAL - ISIDE
      ITEST = ABS(NINT(VAL)-IV)
      IF ((ITEST.EQ.5).OR.(ITEST.EQ.10).OR.(ITEST.EQ.15).OR.
     *   (ITEST.EQ.20).OR.(NINT(VAL).EQ.-IV)) GOTO 300
      IV = IV + ISIDE
      ITEST = ABS(NINT(VAL)-IV)
      IF ((ITEST.EQ.5).OR.(ITEST.EQ.10).OR.(ITEST.EQ.15).OR.
     *   (ITEST.EQ.20).OR.(NINT(VAL).EQ.-IV)) GOTO 300
      VAL = VAL - ISIDE
300   IDIG = NINT(LOG10(1.0/RTN)) + 1
      IF (ICASE.EQ.1) THEN
         MINOUT = IV*RTN
         MAXOUT = NINT(VAL)*RTN
      ELSE
         MAXOUT = IV*RTN
         MINOUT = NINT(VAL)*RTN
      ENDIF
      RETURN
      END
```

APPENDIX B. NOMENCLATURE

AR	aspect ratio
b	span
c	local chord
$\bar{c}$	average chord
c_r	root chord
c_t	tip chord
C_D	drag coefficient
C_{Di}	induced drag coefficient
C_L	lift coefficient
C_{Li}	ideal lift coefficient
C_m	moment coefficient about the leading edge of an airfoil
Cm	root chord for circulation model
C_M	moment coefficient
C_{Mo}	moment coefficient about the y axis
C_{Mac}	moment coefficient about the aerodynamic center
ΔC_p	pressure difference coefficient
d	grid element inset
D	drag
ΔD_i	section induced drag
$\Delta\vec{F}$	force per unit span
K	vortex drag factor
L	lift

M	moment
r	distance between field and control points
S	planform area
$\overline{V_{eff}}$	local effective velocity
V_{∞}	remote velocity
w	induced velocity
x	chordwise coordinate
x_{ac}	x coordinate of aerodynamic center
X_{cp}	chordwise position of center of pressure
y	spanwise coordinate
z	height coordinate
(x,y)	field point
(x_o,y_o)	control point
α	angle of attack
Γ	circulation
$\Lambda\Gamma$	incremental circulation
Δ	leading edge sweep angle
θ	transformed spanwise coordinate
λ	tangent of leading edge sweep angle in circulation model
λ	taper ratio in VORTEX model
ρ	density
τ	percent wing taper in circulation model
ϕ	transformed chordwise coordinate

control point	position where velocity is evaluated
field point	position of the vortex element
wing shape	surface slope of the wing, including twist from root to tip and camber

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5002	2
3.	LCDR Chris Lansing Holm 264 Andanada Los Alamos, New Mexico 87544	2
4.	Dr. Theodore H. Gawain, Code 67Gn Naval Postgraduate School Monterey, California 93943	1
5.	Dr. Richard M. Howard, Code 67Ho Naval Postgraduate School Monterey, California 93943	1
6.	Dr. Max F. Platzer, Code 67Pl Naval Postgraduate School Monterey, California 93943	1
7.	Superintendent United States Naval Academy Anapolis, Maryland 21402-5000	1
8.	Commanding Officer Naval Aviation Schools Command Bldg. 633, Naval Air Station Pensacola, Florida 32508-5400	1
9.	Chief of Naval Education and Training Naval Air Station Pensacola, Florida 32508	1
10.	Commanding Officer Naval Education and Training Program Development Center Pensacola, Florida 32509-5000	1

11. Commander 1
Space and Naval Warfare Systems Command
Washington, DC 20363-5100

12. Commander 1
Naval Air Systems Command
Naval Air Systems Command Headquarters
Washington, DC 20361-0001

END
DATE
FILMED
1-89
DTIC